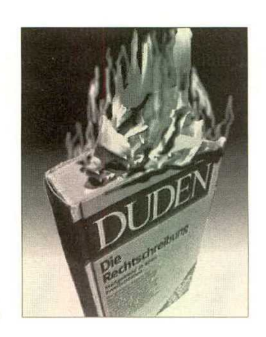

ATARI
ST COMPUTER
Die Fachzeitschrift für ATARI ST- und TT-Anwender
April 91
DM 8,-
Ös. 64,-
Sfr. 8,-
4
Low-Cost-
Laserdrucker
6 Drucker im Vergleich
Phoenix
Datenbank der Zukunft?
Convector
Von Raster-
zur Vektorgrafik
Articolor
Druck
in allen Farben
TT-Manipulation
Problematische ST-Programme
auch auf dem Atari TT
Axel Weigend



Konrad-Duden antWORTET nicht

Die Weisheit, daß Kleider Leute machen, gilt nicht nur für Menschen, sondern auch für Software. Was würden Sie schließlich von einem Editor namens „Editor" halten? Stellen Sie sich vor, Signum! hieße „Schöndruckprogramm mit integrierter Textver arbeitung" - sicherlich kein schöner Name für ein Programm. Doch gerade bei der Namensgebung scheinen viele Software-Häuser große Probleme zu haben, denn die meisten guten Namen sind bereits vergeben. Natürlich könnte Signum! auch „Aphrodite" heißen, aber solcherlei Namensgebung läßt doch einiges zu wünschen übrig, sagt sie doch nicht viel über das Programm aus. So versammelt sich dann die halbe Sagenwelt um den ST und nennt sich Merkur, Wodan, Hermes, Phoenix oder Zerberus.

Doch das Dilemma der Namensgebung nimmt auch recht ungewöhnliche Formen an. Orthografische Ungetüme wie „ReProK", die jedem Redakteur die Tastatur erzittern lassen, sind inzwischen keine Seltenheit mehr. Von Rechtschreibung kann hier keine Rede mehr sein, nur von künstlerischer Freiheit. Hinter dem Namen „OMIKRON." sitzt beispielsweise immer ein Punkt. Das ist zwar werbewirksam, beendet nach „OMIKRON." aber grundsätzlich den Satz, so daß man sich immer überlegen muß, wie man den Namen als letztes Wort im Satz plazieren kann - ganz abgesehen davon, daß das Wort groß geschrieben wird, was orthografisch ebenfalls nicht zu rechtfertigen ist. Hinzu kommen Schöpfungen wie „vortex", „rhothron", „eickmann", „protar" oder als Krönung „mouseWare PAD", die nicht am Anfang eines Satzes stehen sollten, weil sie immer klein geschrieben werden.

Doch nicht nur Software-Hersteller, sondern auch andere Gruppen erfinden immer neue Wortungetüme. Bei Studenten, die ja bekanntlich viel mit Computern arbeiten, hat sich beispielsweise die Bezeichnung „StudentInnen" eingebürgert. Was aus Emanzipationsgründen entstanden ist, sieht inzwischen eher nach dem krampfhaften Versuch aus, eigene Wortschöpfungen zu kreieren. Und selbst hier fehlt eine klare Linie. Geht man davon aus, daß man hieraus die Begriffe „Student", „Studentin", „Studenten" und „Studentinnen" bilden kann, müßte das „Wort" eigentlich „StudentInNEn" heißen!

Ist es nötig, die Rechtschreibung so zu vergewaltigen? Stellen Sie sich eine „frankFurter.SparKASSE" oder den „metzGER_aM_mARKT" vor. Würde Ihnen das gefallen? Der geneigte Leser kann daraus nur eine Lehre ziehen: Nicht alles, was man spricht, ist Deutsh!

Martin Pittelkow

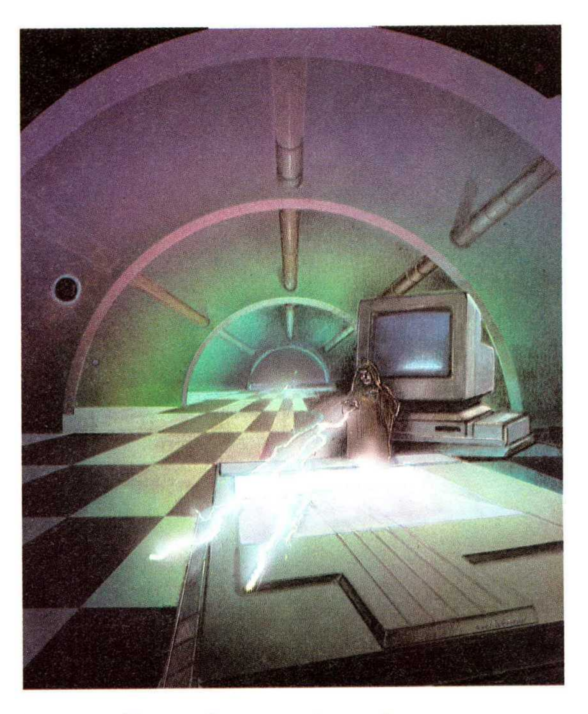
INHALT

SOFTWARE

HARDWARE

ST-REPORT

GRUNDLAGEN

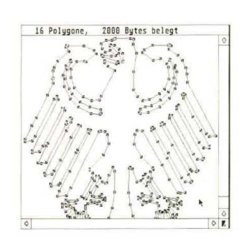


Low-Cost-Laserdrucker

Zu den spannendsten Sachen im Computerbereich gehört, die Verschiebung von Maßstäben zu beobachten. Galt echten Puristen vor Jahren das Druckbild der neun Nadeln noch als adäquat: heute werden sie es kaum noch entziffern können. Stand der Dinge in Sachen Computerdruck sind nun mal - das kann niemand leugnen - die Seitendrucker. Permanent fallende Preise sorgen ebenfalls für Spannung. Wir haben als Einstimmung auf das neue Thema sechs Geräte der unteren Preisklasse untersucht.

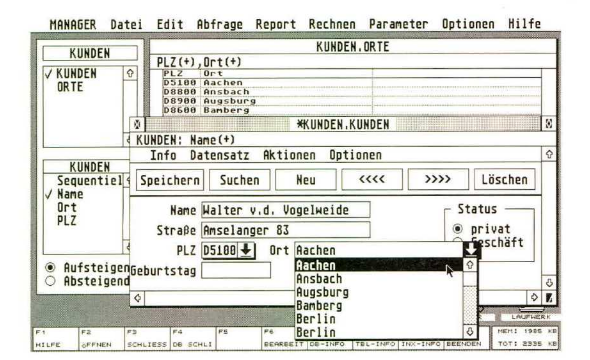
Seite 33

Convector



Die in letzter Zeit immer häufiger auftretenden Programme zur Bearbeitung von Raster- und Vektorgrafiken benötigen zum Austausch von Daten zwischen beiden Grafikarten eine Möglichkeit zur Vektorisierung. Beim Programm Arabesque von SHIFT war diese Umwandlung bisher nur manuell möglich, indem man eine Rastergrafik einfach mit Vektorgrafik nachzeichnete. Für Anwender, denen das zuviel Aufwand war und die lieber ihren Rechner arbeiten lassen (der hat ja auch genug gekostet), bietet SHIFT jetzt in der Reihe 'Arabesque-Tool-Box' ein Programm namens Convector an, welches diese Arbeit erledigen soll.

Seite 20



Phoenix - Datenbank der Zukunft?

Nachdem wir ja bereits in einem Interview in der letzten Ausgabe der ST-Computer einen kleinen Vorgeschmack auf das relationale Datenbanksystem Phoenix bekommen haben, können wir Ihnen hier endlich auch einen Test präsentieren. Mit reichlich Vorschußlorbeeren bedacht, wollen wir Ihnen zeigen, ob Phoenix all' das hält, was es verspricht.

Seite **24**



Articolor

Jeder, der DTP anwendet, ist eigentlich ein kreativer Mensch. Aber seine Produktion endet zumeist mit dem Ergebnis des Laserausdrucks oder der Belichtung. Für viele Kleinaufträge wie Urkunden, Visitenkarten, Poster usw. ist es nicht lohnend, Mehrfarbendrucke im Offset zu fertigen. Genau hier aber setzt das System der Firma Elzner an, um über den konventionellen Bereich (Offset oder Siebdruck) hinaus preiswert farbige DTP-Erzeugnisse zu fertigen.

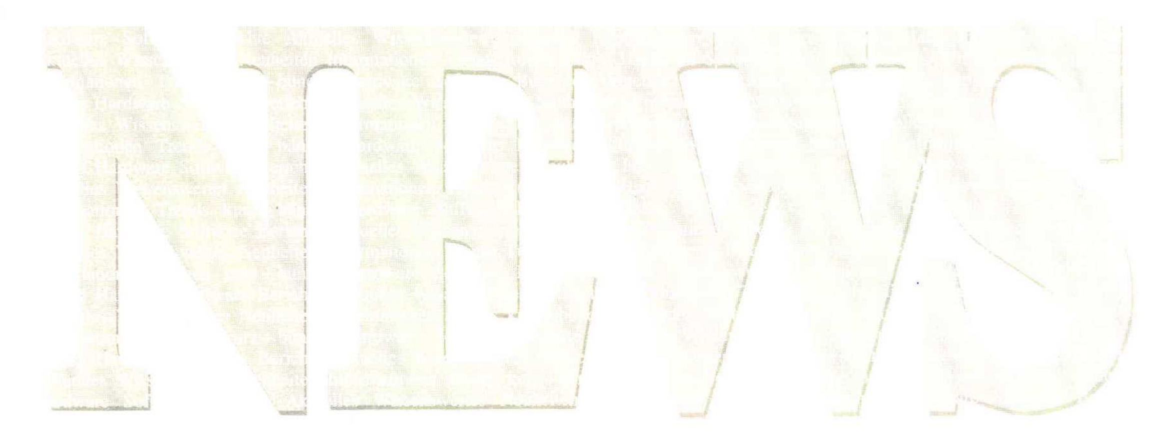
Seite **16**

PROGRAMMIERPRAXIS

AKTUELLES

PUBLIC DOMAIN

RUBRIKEN



2-GB-Laufwerk

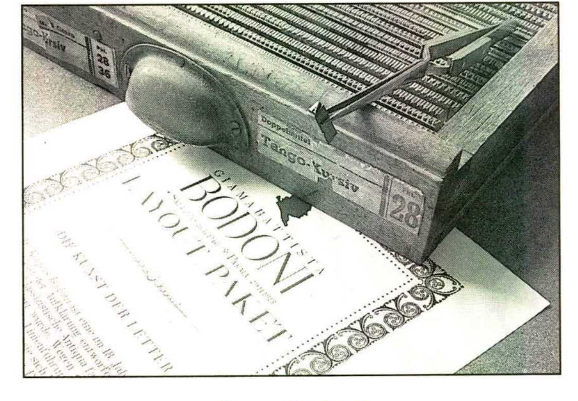
Synelec Datensysteme GmbH bietet ab sofort mit dem DAT-Laufwerk Python von Archive das erste DAT-Laufwerk mit einer Kapazität von 2 GB an. Die Übertragungsgeschwindigkeit von 5 MB/Sek. (Burst) und 183 kB/Sek., SCSI-1 oder SCSI-2-Interface sowie die hohe Kapazität und geringe Bauhöhe zeichnen das Laufwerk aus. Weiterhin hervorzuheben sind die geringe mittlere Suchzeit von 30 Sekunden nach einer beliebigen Datei sowie eine Such- und Rücklaufgeschwindigkeit in der 200fachen Aufzeichnungsgeschwindigkeit. Erreicht wird dies durch 4 Motoren, die Antriebsriemen und Umschaltmechanismen ersetzen. Der SCSI-Controller erlaubt Hard- oder Software selektierbare Konfiguration in SCSI-1 oder SCSI-2. LSI-Technik und Vollintegration ermöglichen die niedrige Leistungsaufnahme von 9,5 Watt. Die synchrone oder asynchrone Betriebsart wird von dem DAT-Laufwerk automatisch erkannt und unterstützt.

Synelec Datensysteme GmbH
Lindwurmstraße 95
W-8000 München 2
Tel. (089) 5179-0

TommySoftware senkt Preise

Der Berliner Software-Hersteller TommySoftware senkt ab sofort die Preise für seine Produkte der MegaPaint II-Serie deutlich. Die MegaPaint II Professional Bookware-Edition kostet DM 299,- und ist vom Leistungsumfang her identisch mit dem Vorgänger MegaPaint II Professional. Die neue MegaPaint Professional Plus-Version wird mit allen Modulen (außer: TT-Modul) sowie vier Font-Disketten für DM 799,- vertrieben. Nur die Plus-Version ist auch als Studentenversion lieferbar. Der Preis des TT-Moduls wurde ebenfalls gesenkt. Statt DM 399,- zahlen die Kunden nur noch DM 199,-.

TommySoftware
Selchower Straße 32
W-1000 Berlin 44
Tel. (030) 621-4063



Laser-Bodoni

Der beliebte Bodoni-Zeichensatz für Signum! und Script ist jetzt auch in einer Laser-Version erhältlich. Das Bodoni-Layout-Paket enthält insgesamt etwa 5300 Zeichen in 46 Schnitten von 7 bis 22 Punkt, Book, Kursive, Bold, Demibold, Light, Versalien, Kapitälchen, Negativ, Ligaturen usw. Der Preis für das Laser-Paket: DM 198,-.

Ebenfalls neu ist das Bodoni Paket Sprachwissenschaft. Bisher war der Satz sprachwissenschaftlicher Arbeiten eine extrem aufwendige und entsprechend unbezahlbare Aufgabe. Jetzt ist es zum ersten Mal auf dem ST möglich geworden, seine Arbeiten komfortabel selbst reprofähig zu edieren. Das Paket enthält 27 spezielle wissenschaftliche Zeichensätze und Grafiken. Darunter befinden sich alle indogermanischen Zeichen, griechische und kyrillische Fonts in mehreren Größen und als Kursive, Phonetik-Zeichensätze, linguistische Zeichensätze usw. Der Preis für das Sprachwissenschaftspaket (Laser und 24-Nadler) beträgt DM 298,-.

Semiotic Soft
Richildenstraße 24
W-8000 München 19
Tel. (089) 174587

VME-Karten für Atari

Ataris neue TT- und Mega STE-Rechner sind serienmäßig mit einem VME-Bus-Steckplatz ausgerüstet, der ohne Öffnen des Gehäuses zugänglich ist. Hierfür bietet die Mainzer Firma PAM Software ihr Ethernet-Netzwerk (10 MBit/s) jetzt auch in der Variante PAM's NET/VME an. In Entwicklung ist noch ein Modul, bestehend aus zwei VME-Karten zur Ankopplung eines Standard-VME-Systems mit mehreren Einschüben. Die Atari-Computer wirken dann als Master auf dem externen VME-System. Die mit dem Netzwerk gelieferte umfangreiche Software wurde weiter verbessert und unterstützt - wie bisher - Multi-Server-Betrieb, d.h. es kann auf alle Platten im Netz, insbesondere die in den TT- und Mega STE-Arbeitsplätzen vorhandenen, mit hoher Geschwindigkeit zugegriffen werden. Mischnetze mit den verbreiteten PAM's NET/E-Adaptern für den DMA-Port, PAM's NET/Emega-Karten und den neuen Mega 2/4-LANstations sind möglich. Auf der CeBIT wird die Mainzer Firma den NFS-Einsatz zur UNIX-Anbindung zeigen, welcher auch parallel zur ST/TT-Vernetzung funktioniert. Darüber hinaus soll eine PC-Anbindung über Novell und BSS+ demonstriert werden.

PAM Software
Carl-Zuckmayer-Straße 27
W-6500 Mainz 33
Tel. (06131) 476312

Nikolaistraße 2
8000 München 40

PRINT TECHNIK

Tel.: 0049-89/36 81 97
Fax: 0049-89/39 97 70

HANDY SCANNER PRECISION 400



Ein neuer, besonders gut zu führender Handy Scanner, der

GRAU und LINE-ART

hervorragend darstellt.

200 / 300 / 400 dpi / 32 grau

Dieses Paket kommt mit dem Malprogramm „Roger Paint" und mit Druckertreibern bis zum ST Laser.

HANDY SCANNER 105 mm DM 498,-
MIT OCR DM 598,-

METEO-SAT-EMPFANGSANLAGE



von der Antenne bis zum Computer incl. Programm. Sie empfangen wie im Fernsehen Bilder vom Satelliten. Ideal für alle, die ständig über das aktuelle Wetter informiert werden möchten.

METEO-SAT MIT FILM SOFT, komplett DM 2.498,-

Videodigitizer PRO 8900 für ATARI



Der Videodigitizer PRO 8805 liefert die höchste Auflösung, die bei Verwendung einer normalen Videokamera möglich ist: 1024 Punkte in 512 Zeilen. Gleichzeitig digitalisiert er mit einer Genauigkeit von 7 bit, was einer Anzahl von 128 Graustufen entspricht. Technische Daten des PRO 8900: Bildformate: Neochrome, IMG, Doodle, Stad, Ausdruck auf: NEC P6/P7. ATARI Laser. Auflösung: 320 x 200, 640 x 200, 640 x 400, 512 x 512, 1024 x 512. Graustufen: 128 (7 bit). Anschluß: ROM-Port des ATARI ST. Eingangssignal: BAS oder FBAS. S/W und Farbmonitor

DM 498,-

Neue Colorsoft von Imagic
16 Farben aus 4096/Zusatzsoft zum PRO 8900 DM 98,-

PRO 8900 mit RGB-Filter + Imagic Soft
Der „Farb-Digitizer" DM 698,-

Realtizer für ATARI ST

Der REALTIZER ist ein in den ROM-Port einsteckbares Modul zur rasanten Digitalisierung von Videobildern aller Art. Die Auflösung beträgt 320 x 200 Punkte, wobei der Farb- und Monochrom-Modus (640 x 400) des ATARI ST unterstützt wird. Die Auflösung: 16 Graustufen. Pro Graustufe beträgt die Digitalisierungszeit 1/25 Sekunde.
Automatische Helligkeits- und Kontrastregelung DM 148,-

RGB-Splitter

Der RGB-SW-Splitter zerlegt jedes Farb-Videosignal in seine Grundfarben Rot, Grün und Blau. Mittels Drehschalter kann jede Grundfarbe mit Schwarz/Weiß an einen Videoausgang geschaltet werden. Passend für alle Videodigitizer mit Farbdigitalisierungs-Software (z.B. PRO 8805).
Noch nie erreichte Farbbildqualität.

DM 198,-

Videotext-Decoder

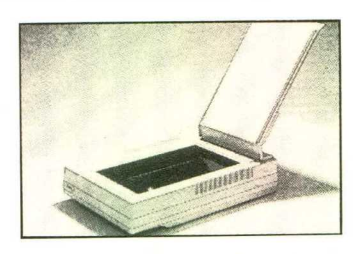
Neue Generation DM 198,-

Zum Anschluß an den ROM-Port. Kann mit jedem Videosignal betrieben werden. Läuft auf Farb- oder S/W-Monitor. Seitenweises Aufrufen – automatisches Blättern – Seiten halten – Speichern und Laden der empfangenen Seiten im Text- oder Bildschirmformat – Textausdruck-Möglichkeit über beliebige Drucker.

Neuer Superpreis/Neue Software DM 1.998,-

PROFESSIONAL SCANNER II

mit OCR-Junior inkl. Ganzseiten-Malprogramm ROGER PAINT OCR Junior, selbstlernende Schrifterkennung,
300 x 300, 300 x 600, 600 x 600 DPI-Auflösung und 64 Graustufen,
Diese Scannereinheit für den Indurstrie- und DTP-Bereich stellt einen absoluten Preishit dar. Mit ihm lassen sich sowohl Halbton als auch binäre Vorlagen scannen und ablegen und mit allen auf dem Markt befindlichen Programmen (auch Calamus) weiterverarbeiten.
Das mitgelieferte Schrifterkennungsprogramm erlaubt das Umsetzen von Text in ASCII-Zeichensatz und ist durch seine Lernfähigkeit von hoher Effizienz



NEU: „NO LIMITS" DIE SUPERSOFT FÜR ST + TT
NEU: DESIGNTE SOFT / WINDOW TECHNIK (8 BILDER GLEICHZEITIG) / SIGNUM-KOMPATIBEL / BLOCKMANIPULATIONEN / IMG / TIF komprimiert-unkomprimiert-grey / STAD-MONOSTAR, SCREEN / POSTER PRINT / SENSATIONELL

UNIVERSAL SCANNER II

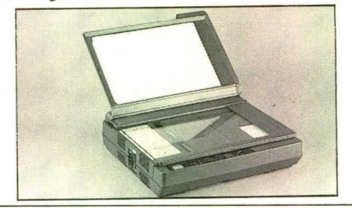
FAX-SCANNER, KOPIERER, PRINTER:

Ein NEUER Universal Scanner löst die alte Generation ab. Endlich mit einem zweiten Motor versehen, stellt er das Gerät wieder in der Ausgangsposition automatisch ab. Mit SuperSoftPaket!

SCAN SOFT / FAXFUNKTION / MALPAKET / OCR 200 dpi / 16 Grau.

Eine Preis-Sensation: DM 1.698,-

NUR FÜR EXPORT oder intern-private Haustelefonanlagen. Ein Anschluß an das öffentliche Telefonnetz der Deutschen Bundespost ist in der BRD nach § 15 Fernmeldeanlagen-Gesetz strafbar.



NEU

OMR = OPTICAL MUSIC RECOGNITION

Paketpreis mit Scanner und Software incl. Manual DM 2.298,-

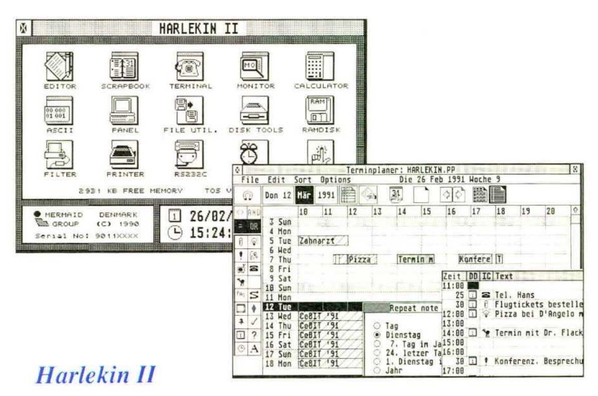
Ein bis zu A4 großes, bedrucktes Notenblatt wird mittels des Print Technik-Universalscanners in den Computer eingelesen. Der Computer verarbeitet das Bild und erkennt die Noten, Pausen, Zeichen etc. Das Musikstück läßt sich über ein MIDI-Keyboard sofort abspielen oder aber abspeichern und mit anderen Programmen weiterverarbeiten, z.B. C-Lab, Notator etc. Erkannte Symbole: Notensystem, Taktstriche, Taktbezeichnungen, G-Schlüssel, F-Schlüssel, Vorzeichen, alle Arten von Noten + Pausen, ganze bis 1/16 Noten, jeder Akkord, Kreuze, BE, Normal, Punkte, Doppelpunkte; Stakkato + Triolen etc.

Wir zeigen diese Neuentwicklung in Hannover!

Wir sind auf der CEBIT: Halle 20, Stand B 05/1

VISA/Eurocard Accepted – Austria: Print-Technik Wien 00 43-222-5 97 34 23

Wir sind auf der CEBIT: Halle 20, Stand B 05/1



Harlekin II

Das Multi-Accessory Harlekin hat sich in der neuesten Version 2.0 zu einer Kombination aus Datenbank, Termin-Manager und Editor gemausert. Natürlich sind alle bisher vorhandenen Features wie z.B. Disketten- uns Speichermonitor, Kopierprogramm, Makros etc. auch weiterhin vorhanden. Harlekin II verfügt jetzt über eine dynamische, reset-feste Speicherverwaltung, so daß immer nur soviel Platz beansprucht wird, wie gerade benötigt wird. Und auch nach einem Absturz hat man in der Regel noch seine Daten. Ein Teil der Harlekin-Module werden bei Bedarf nachgeladen, wodurch der Speicherbedarf stark reduziert wird. Eigene Module, ähnlich den CPX-Modulen des neuen Atari-Kontrollfeldes, lassen sich in Harlekin II einbinden und dann über Tastaturkürzel aufrufen. Viele Harlekin II-Module können jetzt auch aus dem Hauptprogramm konfiguriert werden. Der integrierte Personal Planner wurde stark überarbeitet, so daß er jetzt u.a. an wichtige Termine erinnert, auch wenn sich diese jede Woche wiederholen sollten. Übersichtliche Tages- und Monatsübersichten, Gliederung von Daten und Terminen nach bestimmten Gruppen über Symbole, Suche nach Texten in der Datenbank u.v.m. bewältigt er mit Leichtigkeit. Ausdrucke in vielen übersichtlichen Formen sind möglich (siehe Bild). Auch im File-Utility hat sich etwas getan: In Harlekin II lassen sich jetzt komplette Ordner kopieren. Durch eine schnelle Dateisuchoption findet man ohne Probleme seine Daten und Programme. In der Fileselectorbox wird automatisch auf den richtigen Dateinamen gesprungen. Beim Terminal-Programm wurde u.a. die Kommunikation verbessert bzw. automatisiert. Harlekin II ist zum Preis von DM 159,- erhältlich. Registrierte Benutzer erhalten ein Upgrade zum Preis von DM 60,- und Einsendung der Originaldiskette.

MAXON Computer GmbH
Schwalbacher Str. 52
W-6236 Eschborn
Tel. (06196) 481811

Neue Speichererweiterung von Catch

Eine neuartige Speichererweiterung auf 4 MB stellt die Firma Catch Computer vor. Durch die Verwendung von zwei programmierten Logik-ICs ist es möglich, den im Atari bereits vorhandenen Hauptspeicher mitzunutzen. Einzige Voraussetzung ist, daß dieser mindestens 1 MB beträgt. Da dadurch auf der Erweiterung CCMB4 nur 3 MB nötig sind, kann diese für DM 488,- angeboten werden. Auf der Platine befinden sich zur Erhöhung der Betriebssicherheit zusätzlich Treiber-ICs für die Adreßleitungen. Durch CMOS-Technologie liegt die Leistungsaufnahme bei durchschnittlich zwei Watt.

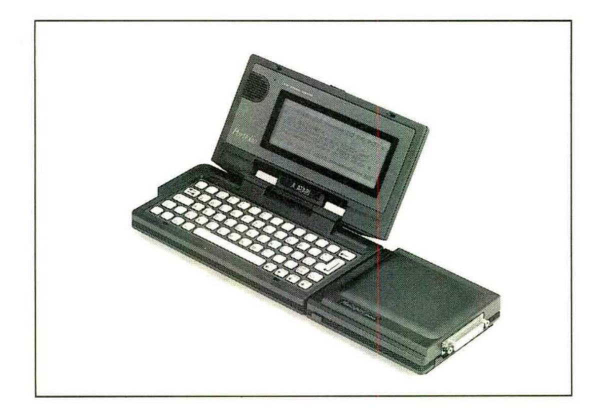
Catch Computer GbR
Ludwigsallee 1b
W-5100 Aachen
Tel. (0241) 157393

Sekretär

Mit dem Programm „Sekretär", das zur CeBIT vorgestellt wird, hat man Serienfax und komfortable Adreßdatenbank direkt aus dem Computer. Weiterhin bietet Sekretär Textbausteine, Gebührenzähler, Paßwortschutz und Verschlüsselung (auch im Export). FAXen und Telefonwahl kann man mit jedem Modem mit Sierra-Chipsatz. Mit FAX kostet das Programm DM 359,-, mit dem passenden Modem DM 669,-. Neu ist auch das Programm PARC, eine digitale Datenbank, mit der man Bilder archivieren kann. Die archivierten Bilder lassen sich über Bildnummer, Namen und/oder Stichworte suchen. Letztere sind frei vergebbar. Das Programm liest alle gängigen Atari-Formate und speichert im gepackten Format. Optional sind auch TIFF-Bilder verarbeitbar.

comtex
Gitteweg 3
W-7801 Bollschweil
Tel. (07633) 50784


LIGHTHOUSE
A&G SEXTON GMBH.



Multiport

Mit dem Multiport stellt die IBP Elektronik GmbH ein zweites Interface für den Portfolio vor. Neben dem Portalog, das als Multimeter/Logger-Interface speziell für Meßaufgaben konzipiert ist, soll das Multiport-Interface alle Portfolio-Anwender ansprechen, die die Aufrüstung ihres Rechners mit mehreren Funktionen realisieren wollen. Multiport beinhaltet eine Centronics-Druckerschnittstelle, eine RS232-Schnittstelle, eine 512 kB-Speichererweiterung sowie einen Steckplatz für ein bis zu 256 kB großes EPROM. Die RS232- und Druckerschnittstellen sind vollständig kompatibel zu den von Atari erhältlichen Interfaces. Für die Datenübertragung zu PCs gehört eine Übertragungs-Software für den PC zum Lieferumfang. Die RAM-Erweiterung rüstet den Portfolio auf insgesamt 640 kB auf, die dann je nach Anwenderwunsch als Laufwerk oder als Systemspeicher verwendet werden können. Der Steckplatz für das EPROM wird als Laufwerk B verwaltet und kann wahlweise mit 256- oder 128 kB-Bausteinen bestückt werden. Mit diesem Steckplatz hat der Anwender die Möglichkeit, Programme und Daten resident zu speichern. Das Laufwerk B ist autobootfähig, so daß Treiber eingebunden werden können. Um das Interface dauerhaft mit dem Portfolio zu verbinden, wird eine Montageplatte mitgeliefert. Der Preis für Multiport liegt bei DM 1499,-.

IBP Elektronik GmbH
Lilienthalstraße 13
W-3000 Hannover 1
Tel. (0511) 630963

BTX-Mail jetzt international

GTC teleCommunication GmbH hat mit sofortiger Wirkung im BTX-System der Deutschen Bundespost einen Übergang zu allen kommerziellen internationalen Netzen geschaffen. Jeder Teilnehmer an Bildschirmtext kann nunmehr Mitteilungen an e-mail-Teilnehmer weltweit versenden, sei es nun EUNET, USNet, BitNet, MCI oder auch im neuen X.400-Netz. Damit stehen den deutschen BTX-Teilnehmern weltweit über 20 Millionen e-mail-Partner zur Korrespondenz zur Verfügung. Der Versand einer Nachrichtenseite kostet, unabhängig von der Entfernung, DM 0,90. Dadurch stellt e-mail aus BTX weltweit die günstigste und schnellste Versandform dar. Der internationale e-mail-Dienst für BTX steht auf der Seite *35008024#. Der Empfang von Mitteilungen aus internationalen Netzen ist ebenfalls möglich. Jedoch ist dazu die Mitgliedschaft in einer geschlossenen Benutzergruppe erforderlich, wie auch beim Telexdienst.

GTC TeleCommunication GmbH
Alexanderstraße 79
W-7000 Stuttgart 1
Tel. (0711) 232653

Schön-Schrift in englisch

Sämtliche zwölf Schön-Zeichensätze (also alle Diskettenvarianten) sind ab sofort auch mit englischer Tastenbelegung lieferbar. Damit wird denjenigen Kunden, die beispielsweise in Großbritannien, Holland oder in den skandinavischen Ländern über keinen Atari mit deutscher Tastatur verfügen, die Möglichkeit gegeben, die Schön-Zeichensätze problemlos zu benutzen. Die englische Tastatur unterscheidet sich nämlich keineswegs nur dadurch, daß Z und Y vertauscht sind, sondern auch in der Lage vieler anderer Zeichen (z.B. Anführungszeichen, Klammern, Akzente, %, &, +, -, ? usw.). Auch fehlen bei der englischen Tastatur die Umlaute Ä, Ö, Ü sowie das deutsche „ß". Diese Zeichen sind deshalb an anderer Stelle verfügbar. Die Zeichensätze mit englischer Tastenbelegung sind in denselben Zusammenstellungen (z.B. Diskette 1 oder 2 oder 3 oder Kombi-Diskette 1+2 oder 1+2+3) wahlweise für 24-Nadler oder für Laserdrucker oder für beide Druckertypen erhältlich. Die Preise bleiben unverändert. Das Anweisungsheft kann allerdings nur in deutscher Sprache ausgeliefert werden, da die zu erwartenden kleinen Stückzahlen die hohen Kosten für ein englisches Anweisungsheft nicht rechtfertigen.

Walter E. Schön
Berg-am-Laim-Straße 133a
W-8000 München 80
Tel. (089) 4362231

Neues von Richter

Augur 2.0 durchbricht erstmals die Einschränkungen der Texterkennungen. Ein neuronales Netz übernimmt auf höherer Ebene Funktionen, die bisher der Programmierer für den Anwender erledigt hatte. Neue Eigenschaften wie automatischer Blocksegmenter, Buchstabensplitter und eine nochmalige Geschwindigkeitssteigerung, die jede für sich bei anderen OCR-Systemen als bahnbrechende Eigenschaften deklariert werden, sind in Augur 2.0 selbstverständlich.

Mit dem „ScanTool" lassen sich komfortabel Bilder nachbearbeiten. Für DM 128,- erhält der interessierte Anwender ein Programm, das er täglich beim Scannen benötigt.

Interessant dürfte auch das „Dulek-Tool" sein, mit dem ein Wörterbuch zweifach benutzbar ist. Das wird dann interessant, wenn ein Wörterbuch mit einem speziellen Thema nicht mehr ausreicht. So lassen sich verschiedene Fachthemen in einem Wörterbuch zusammenfassen, ohne gleich ein neues Wörterbuch erstellen zu müssen. Das Dulek-Tool kostet DM 59,-.

Eine sehr bemerkenswerte Neuigkeit dürfte „Imagic Wizard" werden. Wer „PC-Tools" vom PC kennt und das Programm schon lange für seinen ST sucht, kann die Suche jetzt beenden, denn mit Imagic Wizard gibt es endlich PC-Tools ST bzw. ST-Tools. Lediglich einige festplatteninterne Funktionen konnten nicht in das Programm integriert werden. Imagic Wizard kostet DM 89,-.

Ebenfalls im Angebot von Richter ist ein BASIC-Konverter nach C. Die neue Entwicklungsumgebung für den ST/TT konvertiert GFA-BASIC-Programme in C. Dadurch entstehen keine Turn-Around-Zeiten bei der Entwicklung im Interpreter. Die C-Programme können leicht portiert werden durch eine hochoptimierte kompatible Bibliotheksfunktion. Der konvertierte C-Code richtet sich nach dem modernen ANSI-C-Standard. Dadurch sind die Listings gut lesbar und auf andere Computer übertragbar. Der entstandene Code ist außerdem sofort unter C lauffähig. Der ohnehin schon schnelle GFA-BASIC-Quelltext wird dadurch noch schneller. Der Konverter wird ab der CeBIT mit einem 530seitigen Handbuch ausgeliefert und kostet DM 189,-.

H. Richter
Hagener Str. 65
W-5820 Gevelsberg
Tel. (02332) 2706

Vorn wie hinten erfüllt der MEGA STE höchste Ansprüche

Wie Sie es - oder besser ihn - auch betrachten, das ist sicher: der neue ATARI MEGA STE ist im wahrsten Sinne vielseitig.

Schon auf den ersten Blick ist seine Professionalität unverkennbar. Das moderne Design ist auf ergonomische Bedürfnisse abgestimmt. Klare Funktionalität dominiert.

Der Prozessor MC 68000 mit 16 MHz getaktet, die 3,5 Zoll Floppy, und die 48 MB Festplatte - beides serienmäßig - sowie die reiche Ausstattung mit Schnittstellen ermöglichen Computerleistungen, mit denen Sie Ihre Maßstäbe neu setzen können. Die Vorteile merken Sie sofort, wenn Sie mit dem neuen MEGA STE arbeiten.

Sie brauchen selbst bei anspruchsvollsten Anwendungen keine abstrakten Befehle aufs Komma genau einzutippen. Welch eine Erleichterung! Mit der Maus geht vieles viel einfacher. Über Symbole und Begriffe steuern Sie sämtliche Operationen. Übersichtlich und leicht verständlich zugleich.

Der MEGA STE ist mit allem ausgestattet, was für professionelles Arbeiten auf unterschiedlichen Anwendungsgebieten erforderlich ist. Zum Beispiel für Desktop Publishing, für kaufmännische Abläufe wie Buchhaltung, Textverarbeitung, Dokumentation. Datenbanken. In der Forschung genauso wie in der Entwicklung oder in der Musik und Grafik. Um nur einige Beispiele zu nennen.

Mit Anschlüssen ist der MEGA STE besonders üppig ausgestattet. Drehen Sie ihn doch einfach mal um...



Hinten wie vorn hält der MEGA STE mehr als viele andere versprechen

Eine ungewöhnliche Ansicht, die Ungewöhnliches möglich macht.
Die Vielzahl der Schnittstellen war schon immer eine der Stärken von ATARI Computern. Den einen oder den anderen Anschluß werden Sie vielleicht nicht gleich brauchen. Aber es ist beruhigend zu wissen, daß keine kostenaufwendige Erweiterung des Rechners notwendig ist, wenn Sie sie einmal nutzen wollen.
Da ist zum Beispiel der VME-Bus. Konstrukteure und Techniker wissen sofort, was damit alles in Bewegung gesetzt werden kann. Da sind zwei serielle (RS-232 C) und eine parallele Schnittstelle, der ACSI-Bus (Direct Memory Access) sowie der Anschluß für eine zweite Floppy. Die LAN - Schnittstelle verbindet den MEGA STE mit anderen Computern um im Netzverbund zu arbeiten.

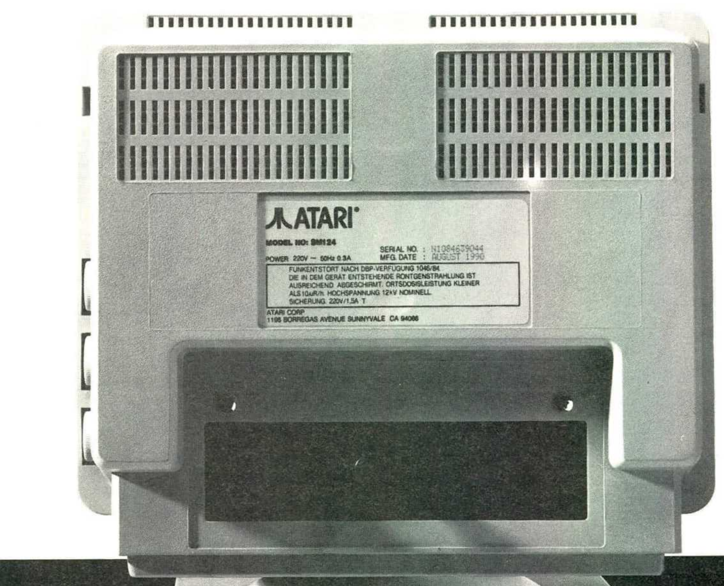
Nicht zu vergessen: TV und Monitor. Die Video-Betriebsarten unterstützen die Auflösungen 320 x 200 Bildpunkte in der niedrigen Stufe, in der mittleren bis höheren Auflösung stehen 640 x 200 und 640 x 400 zur Verfügung. Die Farbpalette reicht bis zu 4096 Farben.
Über den Zwei-Kanal-Audio Ausgang können digitalisierte Klänge analog ausgegeben werden. ATARI Computer gehören seit Jahren zu den meistgekauften in Musikerkreisen. Die MIDI-Schnittstelle hat schon so manchem Musiker zu Weltruhm verholfen.

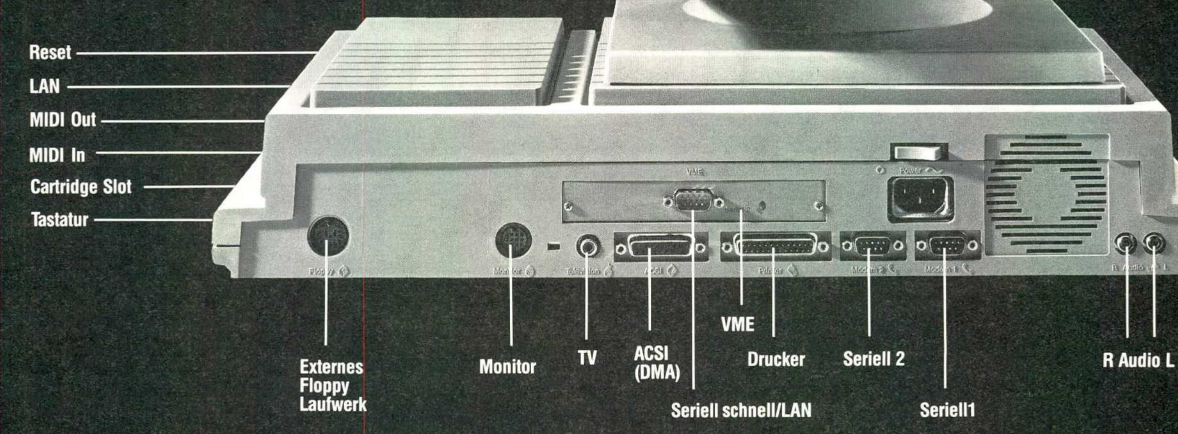
Der neue MEGA STE ist ein Beweis mehr dafür, daß es kein Widerspruch ist, Computertechnologie der Spitzenklasse leicht bedienbar zu machen. Wie einfach das geht?
Drehen Sie den MEGA STE doch einfach mal um...

ATARI®

...wir machen Spitzentechnologie preiswert

Weitere Informationen: ATARI Computer GmbH, Postfach 12 13, 6096 Raunheim





HDPlus 5.02 — **198 DM**
HDPlus ist die speziell für unsere Festplatten entwickelte Treibersoftware mit allem, was zur komfortablen Arbeit mit Massenspeichern und deren Wartung nur vorstellbar ist. Die wichtigsten Funktionen sind einfach zu bedienen, aber auch für den Experten ist HDPlus das universelle Werkzeug. Auf Datensicherheit wurde besonderer Wert gelegt, so können Sie den Rootsektor sichern, Partitionen schreibschützen, oder den Zugriff per Paßwort schützen. Booten verschiedener Accessories von beliebigen Partitions, beliebig viele Partitionen u.V.m.

eickmann Harddisks EX — 30·60·120 MB
Alle eickmann Festplatten werden mit dem neuen HDPlus 5.02 und HDPlus-UTILITIE ausgeliefert. (Fast Filemover von First GbR, Optimizer von Projekt.FPS, Hard Disk Utilitie von Application Systems) Und selbstverständlich anschlußfertig, formatiert, partitioniert, autobootfähig.

z.B.: **EX 60/L** — **1598 DM**
24 ms Zugriffzeit, extrem leise, Autopark

EX 120/L — **2498 DM**
24/24 ms Doppellaufwerk, extrem leise, Autopark

Minidrive Festplatten — 40·60·75·80·100·200 MB
Die schnellen SCSI-Platten im Mini-Gehäuse mit der starken Leistung. Hardwaremäßiger Schreibschutz. Die eickmann Mini Drives wurden gezielt auf Platzeinsparung und freie Plazierungsmöglichkeiten hin konzipiert. Das Gehäuse ist im Design der Mega-Serie gehalten, aber kaum halb so groß!

z.B.: **Minidrive 60** — **1498 DM**
24 ms Zugriffzeit, Single-Laufwerk, SCSI, Autopark

Minidrive 200 F — **3498 DM**
15 ms Zugriffzeit, Single-Laufwerk, 48K-Cache, SCSI, AP.

Megadrive Einbauplatten für Mega ST

z.B.: **Megadrive 60** — **1398 DM**
24 ms Zugriffzeit, SCSI, Autopark

Megadrive 100 F — **2098 DM**
18 ms Zugriffzeit, 16K-Cache, SCSI, Autopark

Wechselplatte EX 40 W

EX 40 W + 44MB Medium — **1998 DM**
25 ms Zugriffzeit, Wechselplatte

Wechselplatte + Festplatte in einem Gehäuse

z.B.: **EX 40 W/75 F + Medium** — **3398 DM**
25/18 ms Wechselplatte + eingeb. 75 MB Platte, Autopark

EX 40 W/80 + Medium — **3198 DM**
25/24 ms Wechselplatte + eingeb. 80 MB Platte, Autopark

EX 40 W/200 F + Medium — **5098 DM**
25/15 ms Wechselplatte + eingeb. 200 MB Platte, Autopark

eickmann EM 124 Multi — **498 DM**
640x400, 640x200, 320x200 Graustufenmultisync

eickmann FolioTalk — **98 DM**
Interfaceprogramm und Verbindungskabel zwischen Atari ST und Portfolio. Die Übertragungssoftware läuft als Accessory oder GEM-Anwendung und ermöglicht den einfachen und sicheren Datenaustausch zwischen ST und Portfolio. Parallele Schnittstelle erforderlich.

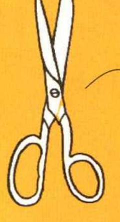
Portfolio Komplettpaket — **598 DM**
Der kleinste PC der Welt. Inkl. Parallel-Interface und FolioTalk (Schnittstelle zum ST)

weitere Angebote und Preise auf Anfrage

ET-der eickmann Tower — Preise auf Anfrage

Der Tower macht Platz auf dem Schreibtisch!
Computer (ST/TT), Festplatte, Wechselplatte, Diskettenlaufwerke, Grafikkarte, Beschleuniger, alternative Betriebssysteme (z.B. Spectre GCR, MS DOS-Emulatoren), Laserinterface, DMA-Buffer, DMA-T-Switch, MS DOS Tastaturmodul, Modem, u.V.m. finden im neuen Gehäuse Platz – unter dem Schreibtisch.
Einfach einschalten und mit der Arbeit beginnen. Auf Ihre zig-fach Steckdosen werden Sie verzichten müssen, denn die Grundkonfiguration, Tower, Bildschirm und Drucker, kommt mit einem Dreifachstecker aus.

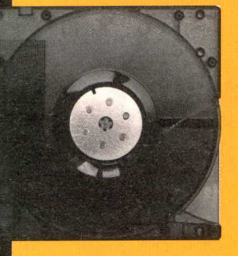






mouseWare DESIGNER MAUS 98 DM
Das optimale Arbeitstier für höchste Ansprüche in den Bereichen DTP, Bildverarbeitung, Grafik und CAD.
Die ergonomisch richtige Form macht die Maus zur sensiblen Fortsetzung der Hand.

►ergonomisch◄ ►schnell◄ ►langlebig◄



mouseWare PAD 19,50 DM
Die Spezialbeschichtung ist genau auf die Gleitflächen der Maus abgestimmt. Mit diesem Pad gleitet die Maus wie auf einem Luftkissen und stoppt exakt dort, wo Sie es wünschen.

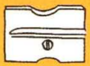
►abwaschbar◄ ►flächenoptimiert◄ ►nahezu unverwüstlich◄

ET - der eickmann Tower

NICHT in den Tower gehören:
Tastatur, Monitor,
Scanner und Drucker.
Ihre gesamte restliche Hardware zieht gern in diesen Tower ein.

- ► vollklimatisiert
- ► ruhige Lage
- ► zentrale Energieversorgung
- ► repräsentative Architektur





Bei der individuellen Ausstattung Ihres eickmann Towers berät Sie kompetent:

Das Planungsteam
von eickmann computer



eickmann computer • In der Römerstadt 249/253
6000 Frankfurt / Main • Telefon 069 / 76 34 09 • Fax: 069 / 7 68 19 71

Neuer Service bei CIS

Die Ciechowski Computer Innovations hat zum Jahreswechsel den Vertrieb und Service für ihr Programm CIS - Lohn & Gehalt auf die CIS GmbH im Wörrstadt übertragen. Für CIS-Anwender ist die Hotline nun montags bis donnerstags von 17 bis 19 Uhr (für Anrufer aus den neuen Ländern von 6 bis 7.30 Uhr) verfügbar. Zusätzlich können CIS-Kunden an einem automatischen Update-Service für ihre Software teilnehmen. Damit ist die Lohnabrechnung immer auf dem aktuellen, gesetzlich vorgeschriebenen Stand.

CIS GmbH
Ober-Saulheimer Straße 18
W-6501 Wörrstadt
Tel. (06723) 7354

PegaFakt 3.0

Die neue Version des Fakturierungsprogramms PegaFakt mit integrierter Lager- und Adreßverwaltung enthält einige neue Features. Unter anderem sind jetzt zum Beispiel Stücklistenverwaltung, Serienbrieffunktion, frei einstellbare Rechnungsformulare, automatisches Schreiben von Bestellungen und automatische Sollberechnung eingebaut. Aber auch mit dem Druck von Paketkarten, zweizeiligen Artikelbezeichnungen, Backup-Funktionen und dem Führen eines Lager- und Kundenarchivs wartet die neue Version auf. Aufgrund des erweiterten Funktionsumfangs erhöht sich der Verkaufspreis von DM 99,- auf nunmehr DM 149,-. Das Upgrade von Version 2.0 auf Verison 3.0 kostet DM 50,-.

PegaSoft
Ringstraße 4
W-7450 Hechingen-Beuren
Tel. (07477) 8158

Neue Version des Dual Standards

Das US Robotics Dual Standard, eines der beliebtesten Modems unter den DFÜ-Freaks, hat jetzt eine neue Platine erhalten. Das Modem beherrscht inzwischen die Standards V.32, V.32bis, V.42, HST (knapp 20000 Baud) sowie die üblichen Übertragungsgeschwindigkeiten 300, 1200 und 2400 Baud. Damit liegt es bislang ungeschlagen an der Spitze. Elbe Datentechnik, der deutsche Generalimporteur für US Robotics-Modems, bietet auf die Geräte 2 Jahre Garantie sowie ein kostenloses ROM-Update, solange das Gerät auf dem Markt ist. Reparaturen werden, sofern notwendig, innerhalb von 3 bis 4 Tagen durchgeführt. Ein HST-Modem kostet DM 1550,- (für Systembetreiber DM 1450,-), das Dual Standard kostet DM 2120,- (für Systembetreiber DM 1990,-).

Elbe Datentechnik
Wittekamp 16
W-3000 Hannover 1
Tel. (0511) 392275

AT-Speed C16

Neu von Sack kommt zur CeBIT der AT-Speed C16. Er verfügt über einen mit 16 MHz getakteten 80286-Prozessor. Dadurch erreicht das Gerät einen 8,2-Norton-Faktor Der EGA-/VGA-Monochrommodus kann emuliert werden. AT-Speed C16 besitzt außerdem einen Steckplatz für einen mathematischen Coprozessor 80C287. Zusätzlich ist das leistungsfähige Betriebssystem DR DOS 5.0 im Lieferumfang enthalten.

Heim Verlag
Heidelberger Landstr. 194
W-6100 Darmstadt 13
Tel. (06151) 56057

Station-One-Creator

Das neueste Programm von Dr. Böhm ist der „Station-One-Creator“, eine Musik-Workstation. Das Programm übernimmt übersichtlich und komfortabel die wesentlichen Editor-Sequenzen beim Erstellen neuer Sounds, Presets, Rhythmen oder Sequenzen der Station One von Böhm. Man erhält sofort einen Überblick über die Daten eines Tracks (Sounds, Presets, Rhythmen, Instrumente, Noten usw.), die direkt mit sofortiger akustischer Kontrolle verändert werden können. Darüber hinaus bietet der Station-One-Creator noch viele weitere Highlights wie z.B. die Darstellung von 24 Tracks (von 48) auf einen Blick, die Edierung aller 800 werkseitigen Sounds (sogar in laufender Sequenz), Wandlung beliebiger Sequenzen in Rhythmen, Intros, Fills, Breaks oder Endings, Speichern von Sequenzen und Rhythmus-Teilen inkl. Drum-Instrumenten auf Diskette und schnelles Selektieren und Löschen aller überflüssigen Daten einer Sequenz.
Als interessante Ergänzung bietet die Firma Dr. Böhm zusätzlich noch ein Konvertierungsprogramm für MIDI-Standard-File-Formate an, den „MIDI-File-Converter“. Mit Hilfe dieses sogar auf dem TT in Farbe lauffähigen Programms lassen sich nahezu alle auf dem Markt angebotenen Songs im Standard-MIDI-Format direkt in die Station One „eindumpen“ oder in Station One erstellte Songs in MIDI-Standard-Files wandeln. Das Programm kann im übrigen auch für das Multifunktions-Keyboard MD 900 T von Böhm verwendet werden.
Für alle Drum-Computer von Böhm in 24-Bit-Technologie werden jetzt weitere neue Drum-Instrumente in Form eines nachrüstbaren 8-Megabit-Chips angeboten, so daß jetzt über 100 gesamplete PCM-Sounds zur Verfügung stehen. Das gilt auch für das Drum-Set DS 16, einen preiswerten Drumsound-Expander in Studioqualität, der insbesondere für den Einsatz in Zusammenhang mit Computern entwickelt wurde. Für den ST bietet Böhm hier einen passenden Drum-Editor an, der jetzt auch die Verwaltung von bis zu 127 Drum-Sounds erweitert worden ist. Mittels dieses Programms können studioreife Rhythmus-Patterns mit Intros, Endings, Fills und Breaks erstellt werden, die sich über ein zusätzliches „Drum-Play“-Programm komfortabel, ähnlich wie bei einem Drum-Computer, verwalten und bedienen lassen.

Dr. Böhm GmbH + Co. KG
Kuhlenstraße 130-132
W-4950 Minden
Tel. (0571) 50450

MegaFakt

MegaFakt, ein Fakturierungsprogramm, ist ab sofort über den Fachhandel erhältlich. Das Programm läuft in der hohen Auflösung des ST, in der mittleren und der hohen TT-Auflösung. Das Handbuch liegt nunmehr in einer überarbeiteten Form vor, außerdem können ab sofort alle Formulare bequem vom Anwender eingestellt werden (mit Wordplus). Der Einführungspreis beträgt DM 1198,- (befristet bis zum 31.03.91), danach DM 1498,-.

MegaTeam OHG
Rathausstraße 1-3
W-5650 Solingen
Tel. (0212) 45888

TT-Monitor

Welcher TT-Anwender hat es sich nicht gewünscht: Statt der zwei Monitore auf dem Schreibtisch nur noch einen einzigen - für alles! Ein deutlicher Platzgewinn. Von CSA DTP Center ist ein 21"-Graustufen-Monitor mit einem Spezialkabel nebst einer speziellen Switchbox erhältlich. Nicht nur die höchste monochrome Auflösung (1280x960 Pixel) steht zur Verfügung, sondern auch 256 Farben als Graustufen in den unteren fünf Auflösungen des TT.

CSA DTP Center
Hüttenstraße 56
4650 Gelsenkirchen
Tel. (0209) 203420





Die neue Dimension für DTP

Jeder, der DTP anwendet, ist eigentlich ein kreativer Mensch. Aber seine Produktion endet zumeist mit dem Ergebnis des Laserausdrucks oder der Belichtung. Genau hier aber setzt das System der Firma Elzner an, um über den konventionellen Bereich hinaus, wie Offset oder Siebdruck, professionelle DTP-Veredlungen auch in kleinen Stückzahlen lohnend zu machen, sowohl zur Zufriedenheit des Kunden als auch lukrativ für den Anwender. Einfach, schnell und sauber.



Unschöne Werbung ...

Zerrissen und zerfetzt hängt ein halbnasses Plakat an einer Hauswand: »oßer Gala-Abend« steht darauf, und dem Betrachter gleitet ein ironisches Lächeln übers Gesicht. Natürlich weiß er, daß es »Großer Gala-Abend« heißen muß, denn er hat gelernt, mit schmuddeligen Publikationen dieser Art zu leben.

Wenn das Plakat schon so aussieht, denkt er sich, kann es mit der Größe dieses Abends ja wohl auch nicht sehr weit her sein. Das triste, schwarz-weiß-grau bedruckte Papier unterstreicht geradezu diesen trübseligen Eindruck. Wenn doch wenigstens ein einziges Fleckchen Farbe auf dem Ding zu erkennen wäre!

Nicht selten liegt es an schlecht gemachter und schlecht verarbeiteter Werbung und weniger am angepriesenen Produkt, wenn der gewünschte Erfolg dann ausbleibt.

Auf derart unzureichende Produkte kann jedoch heute verzichtet werden, vor allem dann, wenn ideenreicher Esprit des grafischen Künstlers sich verbindet mit guter werbetextlicher Aussage sowie einem sauberen und gut gestalteten Erscheinungsbild der Präsentationen.

Auch die Technik stellt kein unüberwindliches Problem mehr dar. DTP hat den herkömmlichen Satz aus vielerlei Gründen schon fast verdrängt und wird ihn auch weiter verdrängen. Das Hauptproblem besteht in der Möglichkeit, preiswert farbige DTP-Erzeugnisse zu fertigen, da die professionellen Farbdrucker noch zu teuer sind.

Für viele Kleinaufträge wie Urkunden, Visitenkarten, Poster usw. ist es nicht lohnend, Mehrfarbendrucke im Offset zu fertigen. Dieses Problem wurde nun gelöst.

Eine neue, farbige Lösung - für DTPler ...

So suchte man in der Druckerei Elzner in Staufen nach einer Lösung und fand sie schließlich auch: Die erste manuelle Farbübertragung für Fotokopierer und Laserprinter. Ein eigens entwickeltes Handgerät mit einer temperaturgesteuerten, heizbaren Walze stellt die Verbindung zwischen den schwarzen Tonerpartikeln und einer darüber gelegten Farbfolie her.

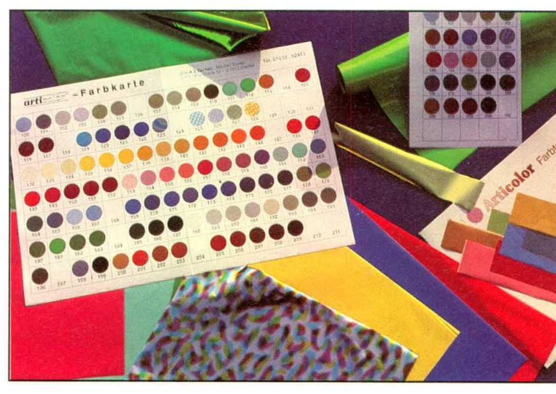
Durch einfaches Überrollen mit der Heißwalze (dem »Articolor Handy-Scooter«) verschmelzen die Tonerteilchen mit der Farbsubstanz der Farbfolie. Durch ein mehrmaliges Abrollen hintereinander können auch größere Flächen eingefärbt werden. Nach Abziehen der Folienteile ist das Ergebnis eine trockene, farbige Reproduktion.

Grafik-Designer und Dekorateure sind begeistert. Endlich können sie Urkunden vergolden oder versilbern, Embleme, Schriftzüge oder Initialen farbig gestalten, ohne auf ihre üblichen flexiblen Gestaltungsmöglichkeiten verzichten zu müssen. Inzwischen gibt es über 100 Farbfolien mit Matt- und Metallic- oder Misch-Effekten, abziehbare Folien zur Übertragung der Farb- oder Schwarzweißelemente auf Papier, Glas, Keramik, Holz, Kunststoff, Metall oder Textilien und anderes mehr.

Farbe ohne Farbdrucker?

Die einzige Voraussetzung sind fotokopierte oder mit Laserprinter ausgegebene Vorlagen. Auch handgeschriebene oder gedruckte Vorlagen können bearbeitet werden. Man fertigt einfach eine Fotokopie von der Vorlage, also auch von Ihren Nadeldruckausdrucken - und schon ist die Grundsubstanz für den Farbtransfer, nämlich die erforderliche Trockentonerschicht, gegeben.

Wer über ein Fotokopiergerät oder einen Laserprinter verfügt, kann diese Geräte benutzen, um zu farbigen Ergebnissen zu gelangen. Der Farbtransfer wird dabei durch die Heißwalze in der Fixierstation des Geräts vollzogen. Die Papiereingabe erfolgt über den Einzelblatteinzug. Mit speziellen Haftklebepunkten werden die Folienteile an jeweils zwei Stellen befestigt, damit sie nicht verrutschen oder im Gerät hängenbleiben. Die Articolor-Farbkarte mit über 100 Farben oder die Desktop-Color-Farbkarte mit ca. 20 Originalfarben bieten eine große Farbvielfalt.



Articolor bietet über 100 verschiedene Farben

Ein Probe-Set für Einsteiger

Für die Direktverarbeitung mit Kopierern und Laserdruckern wird ein Einsteiger-Set angeboten, mit dem unmit telbar die Wirkungsweise des Farbtransfers demonstriert wird und mit dem auch einfachere praxisorientierte Arbeiten realisiert werden können. Ideal für DTPler mit grafischen und typografischen Ambitionen. Professionellere Ergebnisse, gerade auch für vielfarbige oder ganzformatige Übertragungen, werden allerdings mit den speziellentwickelten Geräten erzielt, so mit dem Handy-Scooter mit einer Arbeitsbreite von 40 oder 80 mm sowie einer stufenlos beheizbaren Teflonwalze und elektronischer Temperaturregelung, oder aber mit dem Multifunktionsgerät Arti CTL 320, das nachfolgend beschrieben wird.



Das Multifunktionsgerät Arti CTL 320

Der Markt verlangte im Zuge größerer Formate und größerer Effektivität nach weiteren Lösungen. So wurde ein Gerät entwickelt, das nicht nur in der Lage war, auf einfache und preiswerte Weise und dazu noch sehr umweltschonend aus Schwarzweiß-Farbreproduktionen zu fertigen, sondern auch noch zu laminieren und zu kaschieren und veredeln. Es entstand das Farbtransfer-, Kaschier- und Laminiergerät »Arti CTL 320«, ein Durchlaufgerät für DIN A4- und DIN A3-Formate.

Kaschieren ...

Die Begriffe *Kaschieren* und *Laminieren* sind für viele Interessierte gewiß etwas erklärungsbedürftig. Kaschieren bedeutet soviel wie planmachen, ebnen, ausgleichen. Selbst stark zerknittertes oder eingerolltes oder sehr dünnes Papier wird hierbei auf Karton aufgezogen, also mit einer stabilen Rückwand versehen, so daß ein weitgehend knitterfreies, festgefügtes Produkt entsteht. Ideal zum Beispiel für Preistafeln, Dekorationsschilder, Messe- und Kaufhauswerbung, Displays aller Art und vieles andere, was man gerne aufhängen oder aufstellen möchte, damit es nicht »vom Winde verweht« wird.

Displays oder Dokumente werden auf Karton aufgezogen, indem man einen speziellen Trockenklebeträger zwischen den Karton und das Dokument legt und beides in das Gerät einschiebt, bis die Walzen fassen. Wird das Dokument noch mit einem selbstklebenden Ständer versehen, erhält man sehr rasch ein Display.

Lackieren ...

Wer sein Display nun noch auf Hoch glanz bringen will, verwendet einfach eine Lackfolie. Die Folie wird von ihrem Schutzpapier abgezogen, auf das Dokument gelegt und in den Arti CTL 320 eingelegt. Die hauchdünne Folie bildet durch die Hitze einen Lackfilm. Das Ganze kann noch mit besonderen Effekten versehen werden, zum Beispiel einem Matteffekt mittels Sandpapier oder einer Leinenstruktur, indem man einen Leinenstoff über das Display legt und mit durchlaufen läßt.

Mit der Artiplan Vakuum-Montageplatte in Verbindung mit der Artiplan Vakuum-Thermo-Presse ist es sogar möglich, zerknitterte oder eingerollte Poster wieder zu glätten, fest und sauber einzufärben und zu präsentieren, bis über DIN A1.

Zu all dem gibt es einen Artiplan-Arbeitstisch mit einer mehrschichtigen großformatigen Ablage und der Möglichkeit, Montagearbeiten auf bequeme Weise zu erledigen.

... und Laminieren?

Ganz einfach: Um Speisekarten, Dekorationen, Fotografien, Aushänge, Preislisten, Mitgliedsausweise, Visitenkarten und vieles andere vor Schmutz, Nässe, vor dem Zerknittert- und Zerrissen-werden zu schützen, müssen sie laminiert, das heißt

mit Glanz- oder Mattfolie überzogen oder in diese eingeschweißt werden. Jeder weiß wohl, was damit gemeint ist, wenn er seinen maschinenlesbaren Personalausweis zur Hand nimmt.

Mit den Laminiertaschen, die es von Scheckkartengröße bis zum Format DIN A3 gibt, können Fotos, Farbkopien und Dokumente dauerhaft versiegelt werden, und zwar gleichzeitig von beiden Seiten. Die eingefärbten und versiegelten Dokumente, die auch auf Magnetfolie aufgezogen werden können, sind besonders beliebt als Autoschilder oder Hinweistafeln. Auch das Typen schild des ARTI CTL 320 wurde auf diese Weise hergestellt.

Der Arti CTL 320 ist damit wirklich ein multifunktionales Instrument, mit dem man gleichermaßen Farbe übertragen, kaschieren und laminieren kann. Das Gerät hat eine Arbeitsbreite von 32 cm und ist mit 2 beheizbaren Silikonwalzen für die Eingabe bestückt. Das Eingabematerial läuft über einen Heizkanal und wird von zwei weiteren Silikon walzen wieder aus dem Gerät heraustransportiert. Spezielle Führungstaschen sorgen dabei für eine saubere Transportführung und eine optimale Qualität. Die stufenlose elektronische Geschwindigkeitsregelung und eine elektronische Temperaturregelung bis 180° C ermöglichen auf komfortable Weise die individuelle Handhabung unterschiedlicher Materialien und Formate.

Die vielfältigen weiteren Einsatzmöglichkeiten diese Geräts, die wir aus Platzgründen hier gar nicht alle aufzählen können, machen den Arti CTL 320 zu einem wirklichen Multiplus-Creativ-System und weiten die Einsatzmöglichkeiten des DTP in großem Umfang aus.

Vielfältige Möglichkeiten

Das Multiplus-Creativ-System verfügt natürlich neben den unterschiedlichen Geräten für jeden Zweck über eine Fülle von Zubehör wie Kopier-, Farbfolien, Selbstklebekarton, Magic-Papier zum Abziehen und Aufkleben, Laminiertaschen, Kartonschutztaschen, Abtupf- und Klebeband sowie der Articolor-Farbkarte mit über 100 Originalfarben in echter Handarbeit auf dem Format DIN A3.

Wie bringt man einen roten Elefanten auf einen gelben Hintergrund? Kein Problem: Eine schwarze Volltonfläche wird gelb eingefärbt, ein erneuter Druck oder Fotokopie aufgebracht und rot eingefärbt. Resultat: ein roter Elefant auf einem gelben Hintergrund. Haben Sie noch mehr Ideen?

Mittels der Arti Clear-Folie (einer Klarfolie ohne Farbsubstanz) kann der Toner beim Aufschmelzen eingeglänzt werden. Dadurch werden Schrift oder Bildelemente dichter, was besonders für Reproaufnahmen von Vorteil ist. Außer Transparentpapier kann auch sogenanntes Barytpapier eingesetzt werden, wobei letzteres direkt für die Plattenbelichtung im Offset- bzw. Siebdruck verwendet werden kann.

Die Tonersubstanz wird, z.B. bei PVC-Aufklebern, in das Trägermaterial eingebrannt und damit witterungsresistent und tonerwischfest. Auch für Glückwunsch-, Hochzeits- und Visitenkarten ist dies von Vorteil, ebenso für alle dokumentenechten Anforderungen.

Direkt mit dem Laserdrucker können vorgestanzte Visitenkarten in mehreren Nutzen (durch Vervielfätigung einer Druckvorlage wird eine bessere Ausnutzung des Papierformats einer Druckmaschine erreicht.) direkt eingefärbt und bedruckt werden. Am Ende brauchen sie nur noch ausgebrochen zu werden und erfordern keinen Endbeschnitt. Künftig wird es auch Visitenkarten auf Selbstklebekartons geben. Der Druck wird zunächst einseitig aufgebracht. Nach dem Abziehen kann eine doppelseitig bedruckte Karte entstehen - einfach durch Zusammenkleben der beiden Karten mit unterschiedlicher Information. Auch Karten mit abgerundeten Ecken und verschiedenen Größen, vorgedruckte, mit Streifen oder Rasterverläufen versehene Karten in verschiedenen Farben sind erhältlich.

Was Buchbindereien und Taschenbuch-Fertigstraßen heute im großen erledigen, kann nun jeder auch im kleinen machen:

Mit sogenannten Schmelzklebemappen können Loseblattwerke zu Broschüren gebunden werden. Diese Mappen gibt es in verschiedenen Farben aus Karton oder mit Klarsichtdeckel. Die Mappen werden einzeln in das Arti Multiplus-Gerät, und zwar bei Stillstand der Walzen, gelegt. - Und schon ist aus einem Blätterwald eine hübsche, handliche Broschüre geworden!

Susis Traum: Übertragen mit Magic-Papier

Schon immer wollte Susi ein hübsches rotes Herzchen auf ihrer Schultasche, mit ihrem Namen darunter. Das kann sie jetzt haben - ebenso in ihrem Album, an ihrer Blumenvase und, und, und ...

Namen und Herzchen müssen seitenverkehrt auf das Magic-Papier auf gedruckt werden. Der Toner haftet dabei nur leicht am Übertragungspapier. Nun kann das Kunstwerk auf die gewünschte Stelle übertragen werden. Die Toner teilchen lösen sich durch Andrücken und per Hitze vom Magic-Papier. Mit dem Handy-Scooter einwalzen - und schon ist das Kunstwerk übertragen. Abschließend kann das Herzchen rot eingefärbt werden und der Name mit irgendeiner anderen Farbe. Susi jedenfalls wird glücklich sein. Eine andere Möglichkeit besteht darin, Toner und Folie gemeinsam zu übertragen. Man legt die Farbfolie auf das bedruckte Magic-Papier. Nach der Übertragung haftet der Toner an der Folie und kann als Ganzes transferiert werden. - Bei Glas oder glatten Flächen sollte man die Stelle zuvor mit Zaponlack lackieren.

Darüber hinaus besteht auch die Möglichkeit, einen Aufdruck mit einem Streifen Tesafilm abzunehmen. Damit erhält man schnelle Aufkleber für Kennzeichnungen, Postversand und andere Hinweisschildchen. Sogar Drucke auf T-Shirts lassen sich mit Magic-Papier übertragen. Für professionellere Arbeiten benutzt man dazu jedoch besser eine T-Shirt-Presse.

Zum Beseitigen von Tonerrückständen zur mehrmaligen Verwendung des Magic-Papiers gibt es ein Abtupfband, das auch als Montage-Klebeband für die Farbfolienmontage verwendet werden kann.

Der Kreativität kaum Grenzen gesetzt ist mit der „Arti-Strip" Abziehbildfolie. Und das ist so einfach und macht so viel Spaß wie ehemals die Schiebebildchen aus den Kindertagen.

Nachdem man das Schutzpapier entfernt hat, wird auf die Trägerfolie eine Kopie oder ein Laserdruck reproduziert. Natürlich auch über Farbkopierer. Es können selbstverständlich mehrere Motive auf einer Seite kopiert werden. Anschließend wird das gewünschte Objekt mit einem freien Rand von ca. 2 mm ausgeschnitten und etwa 1 Minute lang in eine Schale mit lauwarmen Wasser gelegt. Das Bild läßt sich nun ganz leicht vom Untergrund abschieben und auf einen beliebigen Gegenstand übertragen. Jetzt wird das Schutzpapier darübergelegt, mit dem Handballen oder einer Rolle darübergestrichen bis keine Blasen mehr zu erkennen sind. Nach 2-3 Stunden ist das Bild fest ausgetrocknet.

Tip: Das Trocknen mit einem Haarfön beschleunigt wesentlich den Trockenvorgang. Sollte das Motiv spülmaschinenfest gemacht werden, müßte man es im Backofen bei ca. 130° C, 10 Minuten einbrennen.

Die matte, selbstklebende Polyesterfolie eignet sich besonders für Aufkleber und Außenbeschriftungen. Sie läßt sich natürlich auch einfärben und bietet viel fältige Einsatzmöglichkeiten.

Zur Abrundung des Multiplus-Creativ-Systems gibt es ein neuartiges Schneidegerät mit einer hochwertigen Edelstahl-

klinge und einer Schnittlänge von 35 cm. Damit können Kartons von einer Stärke bis über 350 g geschnitten werden.

Vergrößern, verkleinern, kopieren

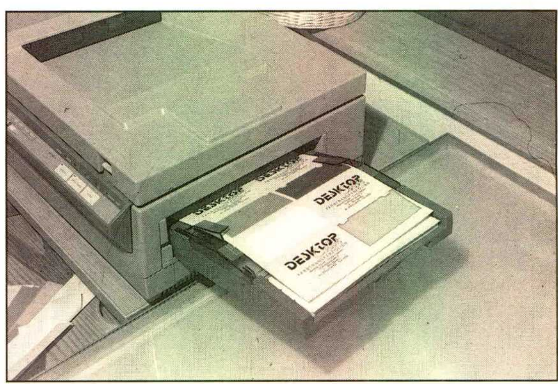
Für den professionellen Einsatz empfiehlt sich ein Großformatkopierer mit einem Format über DIN A2 mit einem stufenlosen Zoom von 50 bis 200% mit 3 verschiedenen Papierkassetten und einem Stapeleinzug. Stellen Sie sich vor, Sie nehmen Ihr DIN A4-Laserdruckdokument und vergrößern es direkt auf eine Plakatgröße von DIN A2 und veredeln es mit Farbe oder Folie.

Für diese große Anwendungspalette wurde speziell eine Transportschutz- und Veredlungstasche entwickelt, die eine besondere Hafteigenschaft bei der Verarbeitung hat, um möglichst optimale, problemlose Ergebnisse zu erzielen. Sie dient somit nicht nur dem Schutz, sondern hat wesentlichen Anteil an der Qualität der ausgeführten Arbeiten. Deswegen sollte man alle Anwendungen (außer evtl. für das Laminieren) damit ausführen. Diese Tasche gibt es in verschiedenen Größen und ist ca. 30-50 mal verwendbar.

Das professionelle Know-how in Fragen des Offset-Drucks, des Schriftsatzes und der Weiterverarbeitung dürften ausreichend Garant dafür sein, daß nicht nur einfach Systeme verkauft werden, sondern daß Dienstleistung und Service in der Druckerei Elzner eine entscheidende Rolle für den Kunden auch in Zukunft spielen werden. Von dem Multiplus-Creativ-System gibt es eine Gesamtübersicht und Bedienungsanleitungen der einzelnen Anwendungsmöglichkeiten.

DK

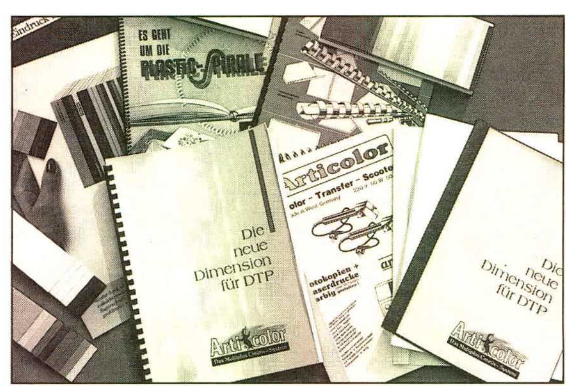
Bezugsquelle:
Manfred Elzner Druck+Sachen
Abt. Articolor-Systeme
Großmattenstraße 12
W-7813 Staufen
Tel. (07633) 82413



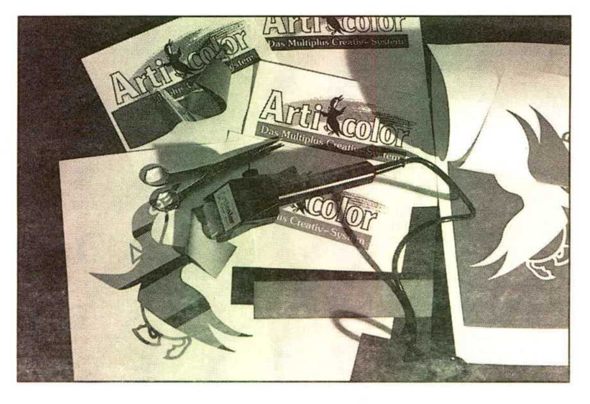
Visitenkartendruck via Laser



Vorgestanzte Visitenkarten (10 Nutzen pro Bogen)



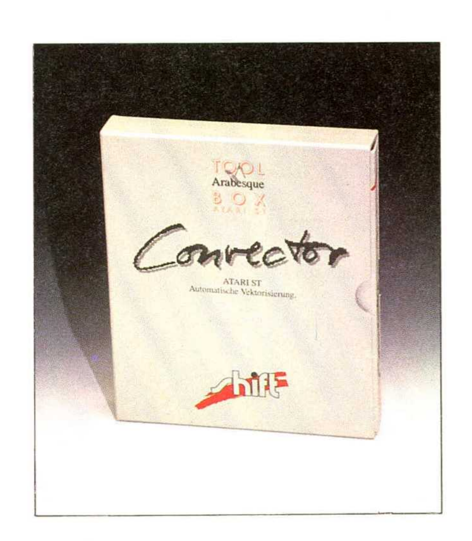
Auch Broschüren lassen sich ansehnlich gestalten



Eine Farbtransfertasche mit dem Articolor-Handy-Scooter

Convector

Ein Programm zur automatischen Vektorisierung



Die in letzter Zeit immer häufiger auftretenden Programme zur Bearbeitung von Raster- und Vektorgrafiken benötigen zum Austausch von Daten zwischen beiden Grafikarten eine Möglichkeit zur Vektorisierung. Beim Programm Arabesque von SHIFT war diese Umwandlung bisher nur manuell möglich, indem man eine Rastergrafik einfach mit Vektorgrafik nachzeichnete.

Für Anwender, denen das zuviel Aufwand war und die lieber ihren Rechner arbeiten lassen (der hat ja auch genug gekostet), bietet SHIFT jetzt in der Reihe 'Arabesque-Tool-Box' ein Programm namens Convector an, welches diese Arbeit erledigen soll. Als Accessory kann Convector mit Arabesque über die sog. Message-Pipe kommunizieren und wird damit quasi zu einem Teil des Hauptprogramms. Aber auch Benutzer, die nicht im Besitz von Arabesque sind, können mit Convector arbeiten.

Convector als Programm

Es erscheint eine Menüleiste und der Hunger nach Rastergrafikdaten äußert sich sogleich in Form einer geöffneten Fileselectorbox. Es werden die Dateiformate IMG (GEM-Image), ABM (Arabesque-Raster), PAC (STAD) und PC3/PI3 (Degas) unterstützt. Es können also nur monochrome Bilder verdaut werden, was man aber nicht als Nachteil bezeichnen kann. Das Bild erscheint dann in einem (im Gegensatz zu Arabesque) richtigen GEM-Fenster und kann ggf. mit Scroll-Pfeilen und -Balken im Fenster verschoben werden.

Sogleich zieht es mich zum Menüpunkt *Vektorisieren*, worauf sich eine Dialogbox im Apple-Macintosh-Design öffnet (Abb. 1). Ich bin nicht unbedingt ein Freund dieser winzigen Mac-Buttons (Grobmotoriker können da manchmal Probleme bekommen...), aber getreu dem Motto 'fast getroffen ist auch vorbei' wird nun erst 'mal auf den Buttons herumgedrückt. Geübte Keyboard-Spieler können die meisten Knöpfe auch über die Tastatur fernsteuern, und zwar mit gedrückter Alternate-Taste und dem unterstrichenen Zeichen. Ach ja, und die 'Flying dialogs' von Julian Reschke wurden auch gleich implementiert, was hier sicherlich sinnvoll ist, falls man sich vor dem Vektorisieren noch ein letztes Mal das Bild ansehen will, dieses aber durch die Dialogbox verdeckt wird.

Vektorisierungs-Parameter HELP
Bitmap: Seite / Block
Toleranz: exakt normal grob
Konturieren Rauschpegel 05 Pixel
Optionen: Umriss / Kontur / fett
Polygone: Optimieren / Füllen / 128 Punkte
Abbruch Start

Abb. 1: Die Dialogbox zum Vektorisieren

Im Rahmen *Bitmap* kann man, falls Convector zusammen mit Arabesque benutzt wird, auswählen, ob die aktuelle Rastergrafikseite oder der Inhalt des Blockpuffers vektorisiert werden sollen. Beim stand-alone-Betrieb ist diese Auswahl gesperrt. Unter *Toleranz* wird eingestellt, wie genau die Vektorisierung vorgenommen werden soll, mit *Konturieren* kann die Grafik vor der Vektorisierung auf ihre Konturen reduziert werden. Ist hier *Umriß* eingestellt, wird eine Umrißlinie gezeichnet, welche um ein Pixel größer ist als das Original; bei *Kontur* ist der Umriß genauso groß wie das Original, und *fett* bestimmt die Dicke der Umrißlinie.

Ist die Rastergrafik ein gescanntes oder digitalisiertes Bild, treten manchmal kleine Verunreinigungen durch Staub oder Schmutz (wer hat da wieder beim Scannen mit Keksen gekrümelt?) im Bild auf, diese können mit einem Rauschfilter eliminiert werden. Alle Krümel- äh Pixel-Gruppen unterhalb des eingestellten Wertes werden dann beim Vektorisieren ignoriert.

Der letzte Rahmen erlaubt einige Einstellungen bezüglich der Behandlung von Polygonen. Das *Optimieren* führt dazu, daß alle Polygone nach ihrer Erzeugung noch einmal abgetastet und eventuell überflüssige Punkte eliminiert werden. *Füllen* stellt die Polygone als schwarz gefüllte Flächen dar. Und dann wäre da noch die Sache mit den Punkten. Manche Programme sind bedingt durch Einschränkungen älterer Versionen des Betriebssystems und des GEM-Metafile-Formats nicht in der Lage, mit mehr als 128 Punkten zu arbeiten. Tritt dieser Fall auf, kann der Default-Wert von

COMPOSITIONEN

Die Textverarbeitung, Das kreative Grafiksystem, Der Publisher

That's Write

Leicht zu erlernen, praxiserprobtes Bedienkonzept, integrierte Referenzkarte, übersichtliche Fensterdarstellung, konfigurierbare Arbeitsoberfläche, Mausbedienung für Einsteiger, vollständige Tastaturbedienung in deutsch für Schnell- und Vielschreiber
Gleichzeitiger Einsatz von bis zu 20 Schriften, druckerinterne und grafische beliebig gemischt (selbstverständlich auch Proportionalschrift), Download möglich. Das That's Write System verarbeitet neben GEM- auch Signum-Schriften.



Internationale Rechtschreibkorrektur/Trennung nach Langenscheidt, Deutsch mit ca. 3 Millionen Worten – Trennung und Korrektur für F, US, GB, NL, E, S, CH, CAN-F als Option erhältlich. Herausragende Silbentrennung, zuschaltbar »nach Wörterbuch« Seitenansichtsfunktionen (PreView) für eine oder mehrere Druckseiten vor Ausdruck. 1:1 oder verkleinert. Grafikeinbindung (kein 640x400 Limit)
Automatische Erstellung eines Stichwort-, Inhalts-, Abbildungs-, Tabellenverzeichnis, ... Fuß- und Endnotenverwaltung (unabhängig!) — Freie Absatz- und Seiten-Layouts — Gliederungsfunktion für Übersicht und schnellstes Anspringen anderer Textbereiche, Kapitel verschieben, Numerieren,...
Mächtige Makrofunktionen, -editor und -anzeige, alle Tasten können mit eigenen Befehlen und Texten nach Bedarf belegt werden.
Rechnen im Text für komplette Kalkulationen, Auf und Abschläge, Abrechnungen, oder tabellenkalkulationsähnliche Funktionen — Formularfunktionen, Ein- und Ausgabe, ...

Vernissage

Vernissage ist ein gut durchdachtes, praxiserprobtes, reichhaltiges und doch leicht bedienbares Kreativ-Werkzeug. Mit Ihrem Computer und Vernissage werden Sie künftig alles bewältigen, was Stift, Pinsel, Airbrush, Konstruktionswerkzeuge und viele andere sinnvolle Gestaltungsmittel zu bieten haben. Der professionelle Anwender ist ebenso wie der engagierte Freizeitkünstler in der Lage, kreative Ideen in kürzester Zeit optimal in sichtbare Erfolge umzusetzen. Es ist sozusagen vorprogrammiert, daß Vernissage zur Arbeitserleichterung und Produktivitätssteigerung beiträgt, auch aus kommerzieller Sicht. Welche Mal und Zeichentechniken Sie als Grafiker, Illustrator, Maler, Karikaturist, Layouter, Art-Director, Designer, Comic-Zeichner u.s.w. auch bevorzugen, Vernissage wird Sie in individueller Art, wie auch im Ausdruck und in der Erhaltung Ihrer künstlerischen Note unterstützen. Die schöpferischen Möglichkeiten, die Ihnen durch Vernissage zur Verfügung stehen, sind nahezu unbegrenzt. Jede Phase der mehrjährigen Programmentwicklung wurde erprobt und spiegelt sich in einer klaren und praxisgerechten Bedienbarkeit wieder, sodaß Sie nie die Übersicht verlieren werden. Gerade der ungeübte Anwender hat Probleme, innerhalb bestimmter Bereiche Stellen zu schraffieren oder auszumalen, ohne das Umfeld zu beschädigen. Vernissage bietet die Möglichkeit, das Übermalen von Rändern und Konturen zu verhindern. Bisher konnten Grafikfunktionen nur auf rechteckige Bereiche begrenzt werden. Dieses Problem wurde folgendermaßen gelöst: Sie definieren Ihre Arbeitsfläche durch markieren verschiedener Konturen der Grafik. Inner- oder außerhalb dieser Konturen können alle Gestaltungsmittel in Echtzeit verwendet werden.

P.P.M.

Ein mächtiges und komplexes Werkzeug zur Bearbeitung und Gestaltung von Text und Grafik. Ob Sie nun Anzeigen oder Plakate, Zeitungen oder Bücher produzieren wollen, für alle diese Tätigkeiten und mehr, bietet Ihnen der Publishing Partner Master eine Vielzahl von Hilfsmitteln.
Publishing Partner differenziert sich von allen auf dem Markt erhältlichen DTP-Programmen nicht nur durch die Vielfalt der Funktionen, sondern auch durch die Verwirklichung einiger völlig neuer Konzepte. Selbstverständlich wurde auf eine einfache und durchsichtige Bedienerführung Wert gelegt. Import vieler Text und Grafikformate (EPSF, GEM, IMG, TIFF,...) deutsche, englische, französische Trennung, Vektorgrafikeditor, beliebiges mischen von Text, Grafik und Vektorgrafik, Textumfluß nach Konturen, drehen, kippen, neigen aller Objekte (Text, Grafik, Vektorgrafik, Textrahmen) Verwendung von Adobe Type 1 Schriften auch auf Bildschirm möglich, viele Textattribute, automatisches und manuelles Unterschneiden (Kerning), Ausdruck auf Grafikfähige Drucker und Geräte mit PostScript. Text und anderen Objekten kann auch Farbe zugewiesen werden. Ausdruck auf Farbdrucker, Color-PostScript oder als Farbseparation, sonst in Graustufen. Überlappungsfunktion erlaubt den Druck im Posterformat über mehrere Druckseiten. Negativ und Spiegeldruck.

512 Punkten entsprechend verringert werden.

Wie geht's denn so?

Der erste Test: das allseits bekannte Wappentier (Abb. 2) soll vektorisiert werden. Ohne langes Ausprobieren wurde das Ergebnis in Abb. 3 sichtbar. Klickt man jetzt direkt in die Vektorgrafik, erscheint ein kleines Popup-Menü, in dem die Einträge *Punkte*, *Füllen* und *Größe* enthalten sind. Ich wähle *Punkte*. Die Vektorgrafik wird daraufhin neu gezeichnet und es werden alle Polygonpunkte dargestellt (Abb. 4). Das sind mir aber entschieden zuviele, hier muß optimiert werden. Es wird also aufs neue vektorisiert, diesmal mit eingeschalteter Polygonoptimierung. Nach einigen Sekunden geduldigen Wartens erscheint wieder die vektorisierte Grafik, welche erheblich weniger Punkte enthält (Abb. 5). Die Optimierung hat sich also gelohnt.

Als nächstes soll ein Schriftzug vektorisiert werden. In Abb. 6 ist das Ergebnis mit unterschiedlicher Genauigkeit und sowohl mit als auch ohne Polygonoptimierung zu besichtigen.

Nun fragt der potentielle Anwender sich, was bringt mir die Optimierung im Alltag? Generell läßt sich sagen, daß eine optimierte Grafik immer etwas ungenauer ist als die nicht optimierte. Soll jedoch eine Nachbearbeitung stattfinden, ist diese bei einem optimierten Bild einfacher, da hier ja wesentlich weniger Punkte existieren. Wo ich gerade beim Thema Nachbearbeitung bin, etwas mehr Funktionen als die oben angesprochenen hätte man hier ruhig implementieren können. Wenn schon die Polygonpunkte dargestellt werden, sollten sie sich doch zumindest mit der Maus verschieben lassen, so daß man die frisch vektorisierte Grafik etwas 'zurechtzupfen' könnte.

Convector als Accessory

Beim Aufruf von Convector aus Arabesque können wahlweise die aktuelle Rastergrafikseite oder der Blockausschnitt vektorisiert werden. Das Ergebnis, nämlich die Vektorgrafik, wird sogleich auf die aktuelle Vektorgrafikseite ausgegeben. Aber auch eine beschränkte Zusammenarbeit mit anderen Programmen ist möglich. Man kann nämlich nach dem Anklicken des *Start*-Buttons einen rechteckigen Bildschirmausschnitt bestimmen, der dann vektorisiert wird und abgespeichert werden kann.

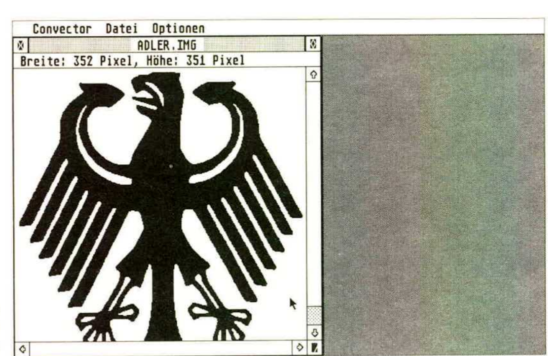


Abb. 2: Zu vektorisierende Vorlage

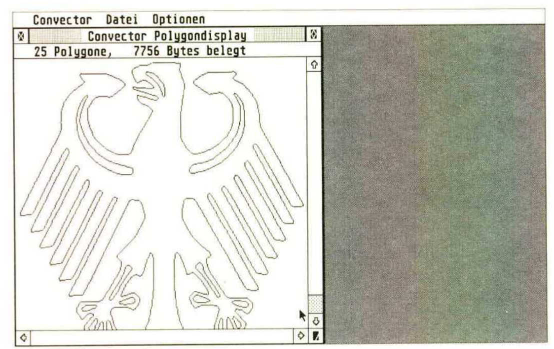


Abb. 3: Vektorisierte Grafik

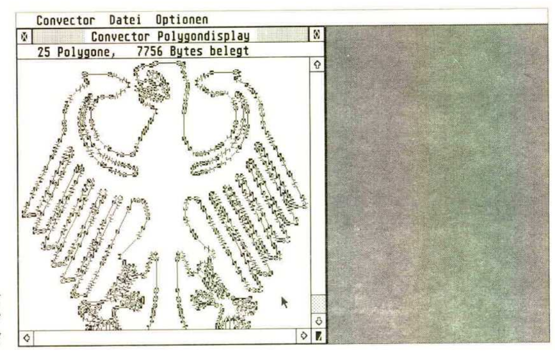


Abb. 4: Darstellung der Polygonpunkte ohne Optimierung

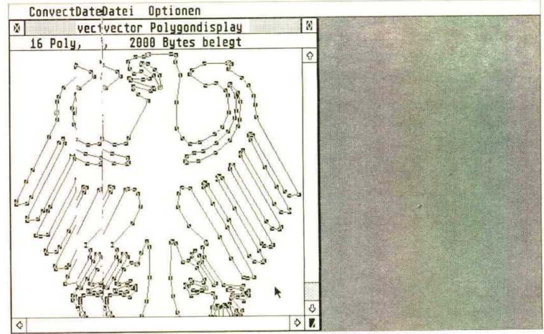


Abb. 5: Polygonpunkte mit Optimierung

Bildfehler

Noch ein Wermutstropfen für Overscan-Besitzer: so ganz kommt Convector mit dieser Bildschirmerweiterung noch nicht klar. Im Accessory-Betrieb werden beim Vektorisieren eines Bildschirmausschnitts die bereits bearbeiteten Punkte gelöscht. Im Overscan-Betrieb treten hier die typischen Symptome für einen Offset-Fehler auf: der Bildschirminhalt erscheint schräg zerrissen. Das gibt natürlich Abzüge in der B-Note! Als Nachtrag zum Arabesque-Test sei noch angemerkt, daß dieses Programm überhaupt nicht mit Overscan klarkommt. Dieses Manko sollte von SHIFT aber schleunigst beseitigt werden - bei einer Professional-Version dürfen solche Fehler nicht mehr auftreten!

Bei dem nur 40seitigen Handbuch wirkt der Ringordner fast etwas übertrieben, aber Ordnung muß schließlich sein. Da die Funktionsfülle des Programms nicht gerade erschlagend ist, reicht dies aber vollkommen aus. Eigentlich habe ich das Handbuch während des Tests kaum gebraucht, das Programm erklärt sich weitgehend von selbst.

Abb. 6: Vektorisierung von Schrift

Fazit

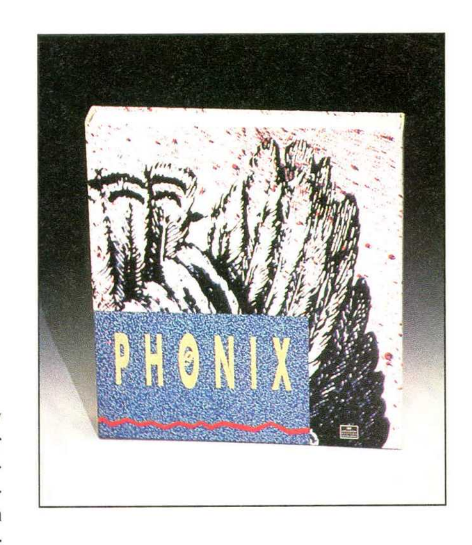
Die Vektorisierung erfolgt recht gut, bei der Polygonoptimierung muß man je nach Grafik etwas warten. Etwas mager sind die Möglichkeiten zum Edieren der Vektorgrafik. Außerdem treten beim Füllen von Polygonen manchmal kleine Fehler auf, aber das Problem haben auch andere Programme diesen Typs. Betrachtet man den geringen Funktionsumfang, der für eine Arabesque-Erweiterung aber sicherlich ausreicht, so bringt einen die unverbindliche Preisempfehlung von DM 248,- doch ins Grübeln. Ein zusätzliches Modul, welches mit zwei Dritteln des Hauptprogrammpreises zu Buche schlägt, wäre mir persönlich zu teuer. Mag auch die Entwicklung eines Algorithmus' zur Vektorisierung nicht gerade zu den einfachen Programmieraufgaben gehören, so ist dies für einen Benutzer doch nur eine Funktion unter vielen anderen zur Manipulation bzw. Konvertierung von Grafiken.

Andreas Hollmann

Bezugsquelle:
SHIFT GmbH
Unterer Lautrupweg 8
W-2390 Flensburg
Tel. (0461) 22828

Phoenix

Mehr als heiße Luft



Nachdem wir ja bereits in einem Interview in der letzten Ausgabe der ST-Computer einen kleinen Vorgeschmack auf das relationale Datenbanksystem Phoenix bekommen haben, können wir Ihnen hier endlich auch einen Test präsentieren. Um es allerdings gleich vorwegzunehmen: Es ist sehr schwer, einem dermaßen umfangreichen Programmpaket wie Phoenix gerecht zu werden, ohne daß irgendein Feature vergessen wird.

Phoenix besteht ähnlich wie andere Datenbanken aus mehreren Modulen. Wer bereits einmal mit anderen Datenbanken gearbeitet hat, dem wird auf den ersten Blick auch gleich alles sehr bekannt vorkommen. Hat man dann noch keinen Blick in das ca. 400seitige Handbuch geworfen und tastet sich in guter alter Testermanier erst einmal im Programm voran, erlebt man immer wieder Überraschungen. Doch auch für denjenigen, der erst einmal nur schnell „reinschnuppern" will, bietet sich im Handbuch ein Quick-Guide, ein „schneller Führer" für Tatendurstige, in dem die ersten Schritte mit Phoenix erklärt werden. Ferner findet sich ein umfangreiches Tutorial am Ende des Handbuchs, in dem Beispieldatenbanken von leicht bis schwierig zu finden sind. Leider fehlt ein Glossar, so daß sich ein blutiger Anfänger zum Thema Datenbank leicht überfordert fühlen kann und zusätzliche Literatur benötigen könnte.

Phoenix verfügt aber auch über ein hierarchisch aufgebautes On-Line-Hilfesystem, das so ziemlich zu allem etwas zu sagen hat. Bekannt ist diese Art von Hilfestellung hauptsächlich aus Programmiersprachen (z.B. Turbo C, MAXON Pascal etc.). Bei Phoenix ist sie aber sogar in Dialogboxen zu finden.

Oberflächlich betrachtet

Eigentlich sind wir jetzt schon mittendrin in der Oberflächengestaltung von Phoenix, die mehr als vorbildlich ist. Es wurden von vielen bewährten Arbeitsoberflächen, die man auf den unterschiedlichsten Computern findet, sinnvolle Komponenten zusammengetragen und unter GEM auf dem ST/TT verwirklicht. Neben dem bereits erwähnten Hilfesystem finden sich Symbolisierung von Tabellen und Vorgängen, Menüleisten in Fenstern, wie man sie vielleicht von Harlekin her kennt, Pop-Up-Menüs für z.B. Datenfelder, verschiebbare Dialogboxen, Tastaturkürzel u.v.m.

Da Phoenix einen eigenen Desktop benutzt, konnten auch Sinnbilder für Drukker, Papierkorb, Laufwerke, Tabellen etc. daraufgelegt werden. Diese Symbolisierung hat den Vorteil, daß man z.B. seine Datentabelle nur noch auf den Drucker zu ziehen braucht, und schon wird sie gedruckt. Fürs Drucken hat Phoenix eine Warteschlange, so daß man mehrere Dateien zum Drucken schicken kann und diese sequentiell abgearbeitet werden. Als Druckertreiber werden die der Application-Textverarbeitung Script genommen, die sich auch leicht verändern lassen, da sie in Form einer reinen ASCII-Datei vorliegen.

Beim Papierkorb sind die Datensätze ähnlich wie auf einem Macintosh nicht gleich gelöscht, sondern man kann noch auf sie zurückgreifen, bis man die Funktion Papierkorb leeren anwählt. Der Papierkorb läßt sich mit einem Doppelklick wieder öffnen, und die „pseudo"-gelöschten Datensätze können betrachtet, Infos darüber geholt, wieder eingefügt oder auch endgültig ins Jenseits geschickt werden.

Doch nun endlich zurück zu den Modulen. Phoenix verfügt über deren zwei: den Manager und den Designer.

Haute Couture

Wenn Sie als Computerbesitzer auch an diesem Genre teilhaben wollen, bietet Ihnen Phoenix ein Designer-Modul, mit dem Sie zwar Schnittbögen vergessen, aber dafür Ihre Datenbank designen können. Immerhin!

Mit dem Designer erstellen Sie Ihre Datenbankstruktur, d.h. alle Masken und grundsätzlichen Einstellungen werden hier vorgenommen. Die meisten lassen sich zwar auch im Nachhinein noch ändern, trotzdem sollte man sich vor dem Erstellen Gedanken machen.

Das Erstellen einer Datenbank erfolgt in einem sogenannten Datenbankfenster. Hier findet man eine Arbeitsfläche und Symbole für die einzelnen Datenbankbestandteile wie Tabelle, Masken, Multi-Indizes etc. Prinzipiell wird eine Datenbank ohne große Tipparbeit mit Hilfe der Maus zusammengebaut. Die Elemente werden einfach aufeinandergezogen. Natürlich geht's nicht ganz ohne die Tastatur, da man den Datenfeldern auch Namen und Längen zuweisen muß. Für Tabellen lassen sich übrigens beliebige Sinnbilder verwenden. Neben den mitgelieferten kann man auch noch selbst erstellte nachladen.

Für Datentypen ist bei Phoenix reichlich gesorgt (s. Bild 2). Neben den „gewohnten" Elementen einer Datenbank wie z.B.

Text, Zahl usw. bietet es auch die Möglichkeit, beliebig große Grafiken im IMG- und GEM-Metafile-Format in eine Datenbank zu integrieren. Will man diese lieber extern verwalten, kann man einen Dateipfad angeben, damit sie nachgeladen werden. Dies hat den Vorteil, daß man die Grafiken mit einem anderen Programm ändern kann und immer die letzte Version in Phoenix hat. Neben den Grafiken lassen sich aber auch gesamplete Musikdaten in Phoenix sowohl extern als auch intern verwalten. Zum internen Einbinden dient das BLOB (Binary Large Object)-Format, das beliebige Daten als Byte, Wort oder Langwort laden kann. Dabei wurde auch an einen Datenaustausch mit PCs gedacht, die einen anderen Aufbau von binären Daten als der Atari voraussetzen.

Über die Feldattribute können Mußfelder, Default-Werte für die Felder, reine Ausgabefelder, Indizes und Austausch von Werten definiert werden. Unter letzterem ist zu verstehen, daß man, wenn zwei Wertetabellen eine Verknüpfung untereinander haben, Werte zwischen diesen Tabellen austauschen kann. Praktisch ist auch die Möglichkeit, Default-Felder anzugeben. Man kann also einen Eintrag in der Maske mit Text und/oder Zahlen vorbelegen. Falls man ein Datum oder die Uhrzeit mit einem Default belegt, erhält man automatisch das Systemdatum bzw. die -zeit beim Eingeben eines Datensatzes. Eine sinnvolle Anwendung wäre z.B., wann eine Bestellung erfolgt ist.

Alle Datentypen einer Datenbank lassen sich auch mit anderen Formaten versehen - sei es beim Datum oder auch bei der Darstellung von Fließkommazahlen, bei denen man z.B. die Anzahl der Nachkommastellen definieren könnte.

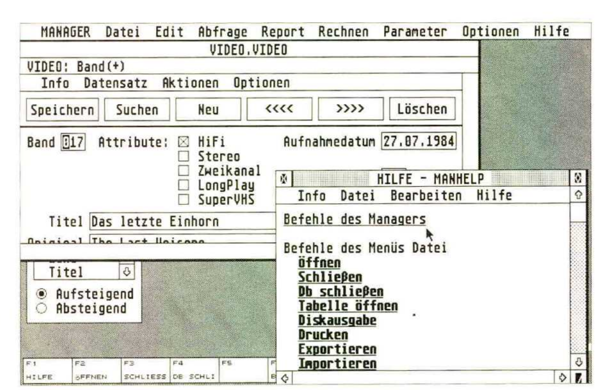


Bild 1: Das On-Line-Help-System von Phoenix

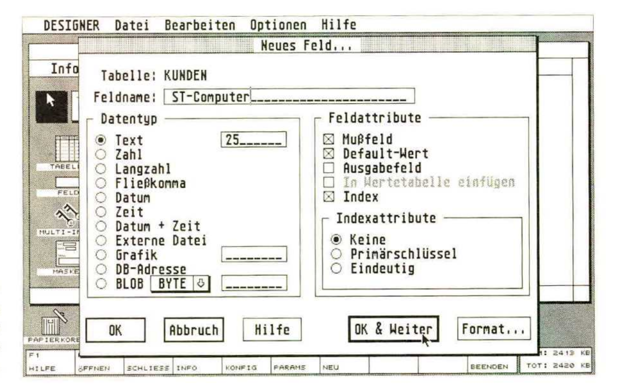


Bild 2: Mögliche Datenformate sind reichlich zu finden.

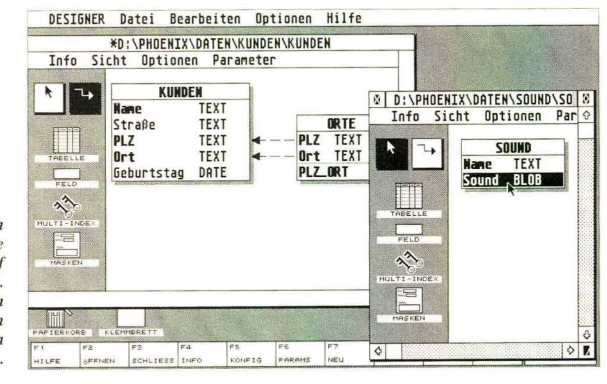


Bild 3: Es lassen sich mehrere Masken auf einmal kreieren. Die Pfeile zeigen Verknüpfungen zwischen Datenfeldern an.

Maskiert

Hat man alle Felder definiert, kann eine Maske erstellt werden. Kann? Richtig, falls gewünscht, nimmt Ihnen Phoenix diese Arbeit ab und kreiert eine Standardmaske. Der wahre Künstler wird aber auf eigene Mittel der Verschönerung zurückgreifen. Dazu lassen sich Datenfelder frei positionieren, Grafiken importieren, Linien und Boxen zeichnen etc.

In Phoenix besteht u.a. auch die Möglichkeit, in seiner Maske Pop-Up-Menüs für Datenfelder, Check-Boxen (Häkchen) oder Radio-Buttons, also Anklickfelder, die sich wechselseitig ausschalten, zu verwenden. Wenn einem die Atari-Systemzeichensätze nicht gefallen, kann man bei Phoenix getrost aufatmen, denn es unterstützt GDOS-Zeichensätze. Allerdings sind weder GDOS noch entsprechende Zeichensätze im Lieferumfang enthalten. Auch stehen eine Palette von 16 Farben zur Verschönerung, diverse Füllmuster, Linien- und Darstellungsarten zur Verfügung - alles, was GEM so bietet. Zur leichteren Positionierung kann ein definierbares Raster eingeblendet werden.

Zuletzt sollte man - sofern man keine Standardmaske nimmt - seine Maske sortieren lassen, da es ansonsten passieren kann, daß der Cursor bei der Eingabe von Daten nicht sequentiell von einem gewünschten Feld zum anderen springt, sondern erst ins erste, dann ins sechste usw. Für die Eingabe von Daten ist so etwas sehr lästig, wie man sich sicherlich denken kann.

Beziehungskiste

Zwischen den Tabellen einer Datenbank lassen sich Beziehungen definieren, d.h. Daten aus einer Tabelle können von Daten aus einer anderen abhängig sein. Man stelle sich ein Reisebüro vor, das in einer Tabelle seine Urlaubsorte hat und in einer anderen die Flugangebote. Die Flugangebote sind von den Urlaubsorten abhängig. Solch eine Beziehung zwischen Ta-

bellen wird einfach mit der Maus hergestellt und automatisch durch einen Pfeil dargestellt. Für diese Art von Verknüpfung lassen sich bei Phoenix auch Regeln für das Einfügen, Ändern und Löschen definieren. Beispielsweise könnte man erreichen, daß, wenn ein Reiseziel wegfällt, gleichzeitig die Flugangebote aus der zweiten Tabelle gestrichen werden.

Es lassen sich aber auch Beziehungen zwischen Wertetabellen herstellen. Z.B. könnte man sich eine Adreßdatenbank vorstellen, die aus einer Adreß- und einer Postleitzahlentabelle besteht. Bei der Eingabe der Adresse in die Adreßtabelle benötigt man nun die richtige Postleitzahl, die aber bereits in der einen Postleitzahlentabelle enthalten ist. Bei Phoenix besteht nun z.B. die Möglichkeit, nur den Ort in der Adreßtabelle einzugeben, und schon hat man die richtige Postleitzahl oder doch zumindest eine eingeschränkte Auswahl, da es mehrere gleichnamige Orte gibt. Voraussetzung ist natürlich die richtige Definition im Designer.

Für die, die es geheimnisvoll mögen, oder z.B. für den Gehaltszettel der Mitarbeiter bietet Phoenix die Möglichkeit, Ihre Datenbank mit Zugriffsrechten zu versehen. Jede Tabelle mit allen Bearbeitungsfunktionen - sogar jedes Datenfeld - kann einzeln gegen unliebsamen Einblick versiegelt werden. Dazu muß allerdings zuerst einmal eine Art Lordsiegelbewahrer bestimmt werden, ohne den dann nichts mehr verändert werden kann. Er vergibt alle Rechte und entscheidet, wer die Leiter rauffällt. Fortan kann man nur noch gegen Losungswort die Datenbank öffnen oder damit arbeiten. Sind bestimmte Felder für einen Benutzer gesperrt, werden sie einfach aus der Maske ausgeblendet bzw. erscheinen erst gar nicht mehr.

Nebenbei läßt sich die Datenbankstruktur als ASCII-Text oder auch als C-Struktur ausgeben. Natürlich kann man eine Datenbankstruktur auch exportieren bzw. importieren, um eine andere Datenbank mit gleicher oder ähnlicher Struktur aufzubauen. Ferner steht im Designer und im Manager ein GEM-Clipboard zum Austausch von Daten mit anderen Programmen zur Verfügung.

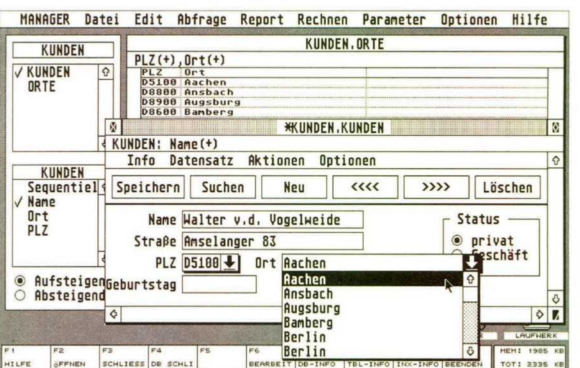


Bild 4: Pop-Up-Menüs und Radio-Buttons in einer Maske sind bei Phoenix nichts Ungewöhnliches.

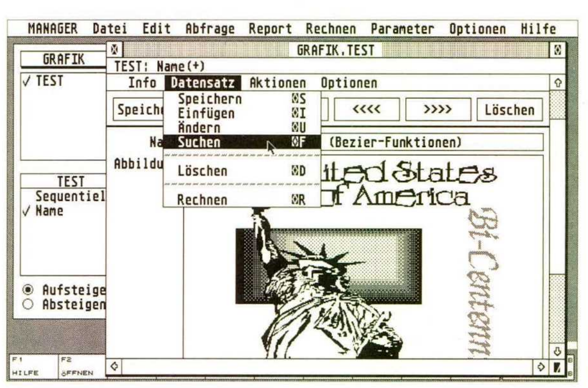


Bild 5: Auch Grafik kann archiviert werden.

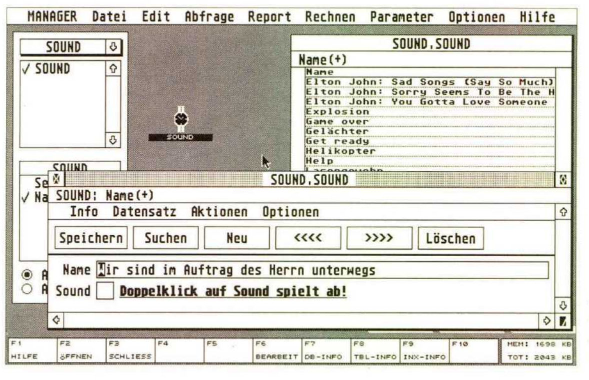


Bild 6: Die akustische Datenbank wird wahr.

Die Kollektion

Nehmen wir das Wort mal wörtlich, dann handelt es sich um eine Sammlung; und genau das ist ja im eigentlichen Sinne eine Datenbank, nämlich eine Ansammlung von Daten. Dazu stellt Ihnen Phoenix gleich einen Manager an die Seite, der Ihre Daten tatkräftig verwaltet.

Mit dem Manager lassen sich Daten erfassen und auswerten. Da ein Multitasking-Kern implementiert ist, kann man bis zu sechs Prozesse gleichzeitig ausführen lassen. Man kann also z.B. Daten einer Tabelle sortieren, während man in einer anderen Daten eingibt. Maximal lassen sich acht Datenbanken gleichzeitig öffnen. Um diese auch verwalten zu können, gibt es auf dem Phoenix-Desktop die sogenannte Datenbank-Toolbox. Hier kann ausgewählt werden, mit welcher Datenbank, welcher Tabelle, welchem Index und ob auf- oder absteigend sortiert werden soll.

Zum Erfassen der Daten gibt es ein Maskenfenster, das neben der eigentlichen im Designer erstellten Maske so ziemlich alles enthält, was man zur Eingabe benötigt; sei es Speichern, Suchen, Löschen oder Hin- und Herblättern zwischen den Datensätzen. Es läßt sich sogar über ein Pop-Up-Menü auswählen, nach welchem Index sortiert die Maske dargestellt werden soll u.v.m. Alle Funktionen sind auch hier über Tastaturkürzel oder eine Fenstermenüleiste zugänglich. Bei der Suche können Wildcards, also Platzhalter für ein oder mehrere Zeichen, verwendet werden. Ebenso kann man die Maske

COLOR WITHOUT PRICE

IMAGINE: Die VGA-Karte für den Mega ST ist da!

IMAGINE ist eine VGA-Karte, die einfach in den freien Slot des Mega ST gesteckt wird. An den Monitorausgang der Karte schließen Sie einen beliebigen VGA-Monitor an.

Auflösungen:

1280 x 960 max.	16 Farben
1024 x 768 max.	256 Farben
800 x 600 max.	256 Farben
640 x 480 max.	256 Farben
320 x 200 max.	256 Farben

IMAGINE wird mit Treibersoftware, GDOS-Treiber und Konverter für nonkonforme Software geliefert. Letzteren werden Sie selten brauchen, hier ein Auszug aus der Liste der von uns getesteten Programme, die problemlos laufen:

Adimens, Arabesque, Calamus, GfA-Basic, LDW Power Calc, Leonardo, Script II, Signum2 !, Technobox Drafter, That's Write, Turbo C, 1st Word plus.

IMAGINE können Sie anschauen. In unserem Systemcenter Regensburg. Oder auf der CeBIT. Sofort!
Natürlich können Sie IMAGINE auch kaufen. Ab Ende April '91.

Für DM 898,–

Wenn Sie noch Fragen haben, rufen Sie uns an. Oder schreiben Sie uns.
Händleranfragen erwünscht.



Aus unserem übrigen Lieferprogramm:

- Atari 520 STE 798,–
- Atari 1040 STFM ... 798,–
- Atari STE 1 MB 888,–
- Atari STE 2 MB ... 1188,–
- Atari STE 4 MB ... 1388,–
- Atari Mega ST 1 998,–
- Atari Mega ST 1 2 MB 1498,–
- Atari Mega ST 1 4 MB 1798,–

- Atari SM 124 298,–
- Atari SC 1224 498,–
- Atari SC 1435 648,–
- VGA Monitore .... a.A.
- AT-Speed 444,–
- Supercharger 666,–
- Atari Megafile 30 ... 698,–
- Atari Megafile 60 . 1198,–

Speicher-erweiterungen:

- ST/STE auf 1 MB 99,–
- STE auf 2 MB 298,–
- ST auf 2,5 MB 448,–
- STE auf 4 MB 598,–
- ST auf 4 MB 748,–
- Atari Portfolio 398,–

- 1st Word plus 3.15 ... 99,–
- Word Perfect 148,–
- Adimens ST plus 198,–
- Aditalk ST plus 248,–
- Turbo C Pro 398,–
- Power Pack 49,–
- Public Domain 5,–
- Weitere Software ...... a.A.

Unverbindlich empfohlene Verkaufspreise

WITTICH COMPUTER GMBH

VERSANDZENTRALE
Tulpenstraße 16 • 8423 Abensberg
Telefon und Fax (09443) 453

LADENVERKAUF
Luitpoldstraße 2 • 8400 Regensburg
Tel. (0941) 562530 • Fax (0941) 562510

24 Stunden Bestellannahme • Telefonische Beratung 10.00 bis 20.00 Uhr

wechseln oder in eine andere oder die gleiche Tabelle verzweigen.

Eine Datenbank läßt sich auch als Liste in einem Fenster anzeigen, wobei ein Mausklick auf einen Datensatz genügt, um das entsprechende Maskenfenster zu erhalten. Hier ist den Programmierern von Phoenix ein Meisterwerk an einfachem Handling gelungen. Will man in dem List-Fenster die einzelnen Datenfelder in einer anderen Reihenfolge angezeigt bekommen, klickt man einfach auf den Datenfeldnamen und zieht ihn mit der Maus an die gewünschte Stelle. Fertig! Oder Sie stellen fest, daß eine Feldbreite von 100 Zeichen für einen Vornamen bei der Anzeige hinderlich ist. Einfach auf die Trennlinie zwischen den Datenfeldern geklickt, und sofort erscheint eine Box, in der man die Breite sieht, während man die Linie einfach verschiebt (s. Bild 7). Will man eine ganz bestimmte Breite, läßt sie sich auch als Zahl eingeben; die Anzeige wird dann sofort geändert.

Doch nicht nur durch das komfortable Handling glänzt Phoenix; durch den eingebauten Cache-Speicher können die Indizes der Datensätze im Speicher gehalten werden, was bewirkt, daß ein extrem schneller Zugriff auf die Daten möglich ist. Die Größe des Caches ist einstellbar, so daß man ihn ganz individuell seinem Rechner anpassen kann.

Bild 7: Die Breite und Reihenfolge von Datenfeldern in einer Liste lassen sich leicht per Maus ändern.

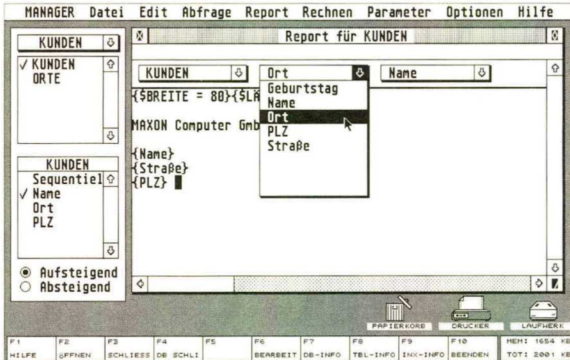


Bild 8: Reports werden mittels Pop-Up-Menüs zusammengestellt.

Fremdgänger

Nun will man natürlich nicht immer nur Daten von Hand eingeben, sondern auch auf bestehende Daten aus anderen Programmen oder Phoenix-Datenbanken zurückgreifen. Dazu gibt es Import-/Export-Möglichkeiten, bei denen sich Datensatztrennung und -umbruch einfach als ASCII-Code einstellen lassen. Probleme können höchstens dann auftauchen, wenn man in der Maske Mußfelder definiert hat und zu importierende Daten dieses Feld nicht ausfüllen. Ist ja auch verständlich! Unter anderem beim Datentransfer begegnet einem das Prozeßfenster von Phoenix. Hier werden Zeitinformationen zum zur Zeit laufenden Prozeß angezeigt, und es besteht die Möglichkeit diesen Prozeß anzuhalten oder für ihn alle Rechenzeit Ihres Atari zu reservieren. Da - wie bereits oben erwähnt - bis zu sechs Prozesse gleichzeitig ablaufen können, und es dann auf dem Desktop etwas eng werden würde - nicht jeder hat einen Großbildschirm -, kann man für jeden Prozeß ein Uhren-Sinnbild auf dem Desktop ablegen. Platzsparend und effektiv! Nun werden Sie vielleicht sagen, Sie wollen nicht sechs verschiedene Dateien gleichzeitig transferieren. Brauchen Sie auch nicht, denn das Prozeßfenster taucht bei allen Prozessen auf, wie z.B. Drucken, Abfrage, Reports etc. Womit wir auch schon beim nächsten Thema sind.

???

Daten in einer Datenbank dürsten nach allen Arten von Abfragen und Weiterverarbeitung. Wollen Sie z.B. aus Ihrer Briefmarken-Datenbank wissen, wie oft sich die „Blaue Mauritius" in Ihrem Besitz befindet, genügen wenige Operationen mit Maus und Tastatur. In einem Fenster, ähnlich dem Listfenster, wird das Suchkriterium definiert und dann die Abfrage gestartet. Daraufhin erhält man je nach Parametereinstellung entweder zuerst eine Meldung über die Anzahl und dann eine Liste der dem Kriterium entsprechenden Datensätze oder gleich letzteres. Bei der Suche können auch wieder Wildcards eingesetzt werden. Die Kriterien können natürlich auch komplexer sein, so daß sie nicht nur die Anzahl Ihrer Briefmarken, sondern auch noch alle Werte über DM 2000,- und abgestempelt erfragen können. Dazu stehen Ihnen diverse Verknüpfungsarten zur Verfügung. Außerdem kann bestimmt werden, welche Felder überhaupt mit in die Abfrageliste aufgenommen werden sollen.

Da bei Phoenix alle Abfragen, Reports etc. aus der Original-Datenbank in diese Liste herauskopiert werden, kann man nach Erstellen beliebig mit ihr herumhantieren. Löscht man sie aus Versehen, kann man sie also ohne Probleme neu erstellen. Ferner werden solche Listen automatisch als temporäre Dateien auf einem Laufwerk ausgelagert. Es empfiehlt sich also die Nutzung einer RAM-Disk dafür. Der Pfad für die temporären Dateien läßt sich einstellen.

Neben den Abfragen lassen sich auch Reporte erstellen (s. Bild 8). Darunter sind formatierte Listen zu verstehen, die man mit Zusatztexten versehen und dann auf Bildschirm, Datei oder Drucker ausgeben kann. Einfache Beispiel sind Adreßaufkleber, Serienbriefe oder Bestandslisten. Man kann dabei über Menüs aus Tabellen, Datenfeldern und -indizes auswählen. Alles wird direkt als Text in einen im Fenster laufenden einfachen Editor eingefügt und kann nachträglich mit Formatmarken wie links- oder rechtsbündig, Anzahl der Zeichen etc. versehen werden.

Ähnlich wie das Report- ist das Rechenfenster aufgebaut. In diesem Teil von

Phoenix können Rechenvorgänge definiert und ausgegeben werden. Es gelten allgemeine Rechenregeln, wie z.B. „Punkt-vor-Strich". Neben den vier Grundrechenarten stehen Klammerung, Systemdatum und -zeit, Nummer des aktuellen Datensatzes usw. zur Verfügung. Mit diesen Möglichkeiten kann man leicht Rechnungen für Kunden schreiben o.ä.

Für Listen kann ein Seitenformat definiert werden, damit nicht Chaos auf dem Druckerpapier herrscht. Dazu bietet Phoenix ebenfalls umfangreiche Einstellmöglichkeiten bis hin zur Summierung von Werten bei Rechnungen.

Alle Abfrage-, Report- und Rechendefinitionen lassen sich abspeichern und jederzeit wieder laden. So muß man eine bestimmte Definition nur einmal erstellen und braucht sich nie mehr neu Gedanken darüber zu machen. Außerdem kann man mehrere Definitionen gleichzeitig im Speicher halten und dann aus einer Liste die gewünschte auswählen.

Auch die Sortierung ist ganz einfach. Man erhält ein Fenster, in dem sich eine Liste der Felder und eine leere Liste befinden. Durch Anklicken in der gewünschten Reihenfolge werden die Felder in die leere Liste übernommen (s. Bild 9). Außerdem läßt sich auf- und absteigend Sortieren und ein Komplement der Sortierreihenfolge erstellen.

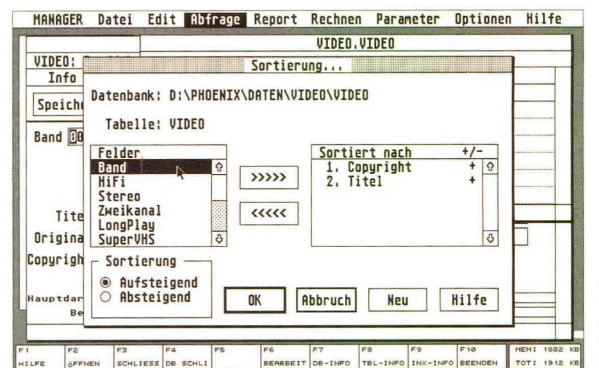


Bild 9: Die Sortierreihenfolge läßt sich ganz einfach bestimmen.

Statt Nadel und Faden

Wie dieses Handwerkszeug kann man das dritte Modul Phoenixs verstehen. Mit ihm kann man - wie der Name Reorganizer schon sagt - eine Datenbank neu organisieren. Das wird z.B. dann nötig, wenn die lieben Kinder mal testen wollten, was denn der Papi macht, wenn man den Stecker des Computers herauszieht, an dem er schon seit Stunden sitzt, statt mit ihnen zu spielen. Hat er gerade Datensätze eingegeben, wird er jubilieren, da seine Daten weg sind. Um nun die Datenbank wieder zu „flicken", muß sie erst einmal reorganisiert werden. Dabei wird überprüft, ob zu Datensätzen Indizes fehlen etc. Auch Löcher, die durch Löschen von Datensätzen entstehen, können auf diese Art und Weise gestopft bzw. die Datensätze zusammengerafft werden.

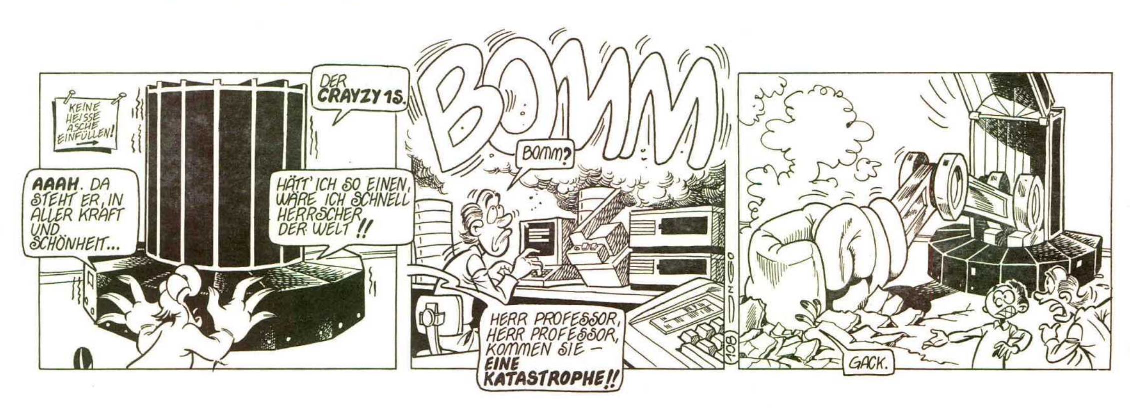
Geschafft!

Wie bereits am Anfang gesagt, ist es sehr schwer, ein Programm wie Phoenix einem Leser näherzubringen und ihm dabei gerecht zu werden. Phoenix bietet einfach zuviel, um es auf ein paar Seiten ausführlich zu beschreiben. Sicher ist jedoch, daß bei einem Preis von DM 399,- ein neuer Meilenstein unter den ST/TT-Datenbanken gesetzt wird, was Bedienungskomfort und Leistung angeht. Wer über einen großen Arbeitsspeicher verfügt, kann Phoenix übrigens auch als Accessory laufen lassen, wodurch es dann aus jedem GEM-Programm über die Menüleiste erreichbar ist. Nicht zu vergessen, daß Phoenix nicht nur auf dem ST/TT ohne Probleme läuft, sondern auch die bekannten Grafikkarten und somit auch Großbildschirme unterstützt.

HE

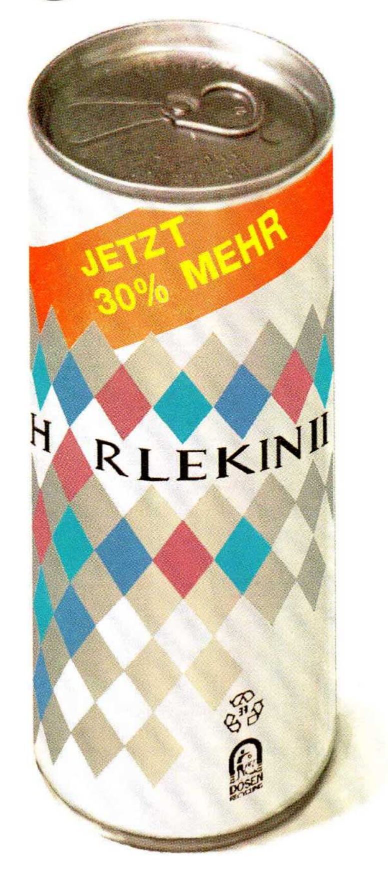
Bezugsadresse:
Application Systems /// Heidelberg
Englerstr. 3
W-6900 Heidelberg
Tel. (06221) 300002





HARLEKIN II

Jetzt mehr Inhalt und weniger Kalorien*



HARLEKIN II ist die revolutionäre Datenbank mit integriertem, stark erweitertem Termin-Manager.

HARLEKIN II ist der schnelle GEM-Texteditor der immer bereit steht.

HARLEKIN II ist das Systemtool, das spoolt, konvertiert, kopiert, formatiert, kalkuliert, makrotisiert, fonteditiert, ramdiskinstalliert und noch dazu zeitoptimiert.

HARLEKIN II ist ständig erreichbar und bietet alles, was Anwender jederzeit brauchen.

HARLEKIN II ist schnell, kompakt und ein wahres Multitalent, das alle anderen Helfer arbeitslos macht.

***Jetzt kalorienarm!**
Unser Motto - mehr Leistung bei weniger Speicherverbrauch. Dazu wurde HARLEKIN II mit einer dynamischen Speicherverwaltung, die noch dazu resetfest ist, ausgestattet.

Jetzt modular!
HARLEKIN II ermöglicht es, einzelne Programmodule nachzuladen. Das gilt für eigene Programmteile und auch für fremde Module, die unter HARLEKIN II gestartet werden können.

Jetzt geschmacksverstärkt!
Erweiterter Termin-Manager, Editor, Disk-Monitor, Terminal, Kontrollfeld, Datei-Manager - TT und Grafikkarten, integrierte Konfiguration, u.v.a.m.

HARLEKIN II
DM 159.- unverbindliche Preisempfehlung

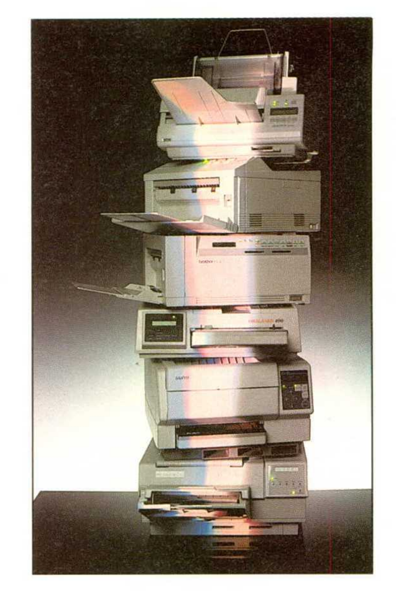
UPGRADE DM 60.-
Gegen Einsendung der Originaldiskette



MAXON Computer GmbH
Schwalbacher Str. 52
W-6236 Eschborn
Telefon (0 61 96) 48 18 11
Fax (0 61 96) 4 18 85

Der Bundesminister für Informatik: HARLEKIN II kann zu extremen Suchterscheinungen führen.
Sonstige Nebenwirkungen unbekannt. Fragen Sie Ihren Fachhandler.
Inhaltsanalyse: hochoptimierter Code folgender Zusammensetzung: Termin-Manager, Datenbank, Texteditor, Disk-, Speicher- und Dateimonitor, DFÜ-Terminalprogramm, Datei-Manager, Taschenrechner, Makro-Maker, Spooler, Druckfilter, RAM-Disk, erweitertes Kontrollfeld, Fonteditor, Iconeditor, Fileselector, Screensaver, Quick-Maus, Kopierprogramm, Systemtools, u.v.a.m.

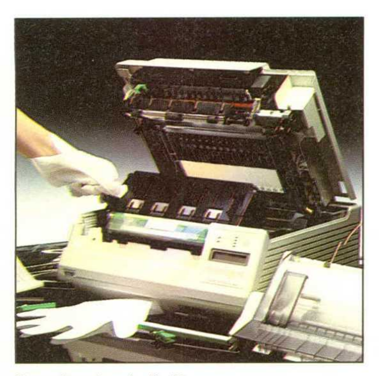
Drucken mit Licht



Sechs Seitendrucker im Vergleich

Zu den spannendsten Sachen im Computerbereich gehört, die Verschiebung von Maßstäben zu beobachten. Galt echten Puristen vor Jahren das Druckbild der neun Nadeln noch als adäquat: heute werden sie es kaum noch entziffern können. Stand der Dinge in Sachen Computerdruck sind nun mal - das kann niemand leugnen - die Seitendrucker. Permanent fallende Preise sorgen ebenfalls für Spannung und dementsprechend sind diese Maschinen keine Utopie mehr. Wir haben als Einstimmung auf das neue Thema sechs Geräte der unteren Preisklasse untersucht.

Ein weiteres Merkmal der Computerbranche sind begriffliche Verallgemeinerungen, die nicht selten herbe Verwirrung schaffen. Da werden Fonts und Zeichensätze verwirbelt, PCs und IBM-Kompatible sind längst schon identisch, wie eben LED-, Ionen-, LCD-, LCS-, kurzerhand zu Laserdruckern erklärt werden.



Vorm Druck steht die Montage: Einsetzen der Entwicklungseinheit beim Seikosha

Das ist aber grundfalsch, genaugenommen handelt es sich bei diesen um 'elektrostatische Seitendrucker'. Zugegeben, 'Laser' ist kürzer, einigen wir uns für die Zukunft auf 'Seitendrucker'. Was sie unterscheidet, ist der Weg, das zum Druck benötigte Licht zu erzeugen. Mehr darüber lesen Sie im Kasten 'Blick ins Innere'.

Marktausschnitt

Wir haben uns auf dem Markt umgesehen und präsentieren Ihnen heute sechs Geräte, deren Preisniveau um bzw. unter 4000 DM liegt. Ein finanzieller Einschnitt droht also immer noch, sollte man sich für eins der Modelle entscheiden. Nichtsdestotrotz handelt es sich bei den Geräten um die Low-Cost-Klasse der Seitendrucker. Vorweg sei bemerkt, daß man natürlich Ataris SLM 605 nicht ignorieren kann, wenn man preiswerte Seitendrucker untersucht. Doch zum einen haben wir das in [1] bereits getan, und zum anderen unterscheidet er sich von unseren heutigen Kandidaten dadurch, daß er durch seinen DMA-Anschluß nur an den Ataris, dort nur mit entsprechender Software, dann aber schnell arbeitet. Daher liefe er außer Konkurrenz, denn die Prüflinge sind ganz normale Drucker, die via Centronics an jeden Rechner angeschlossen werden können.

Bedingung für die Teilnahme war neben dem Preis auch die Kompatibilität zum Hewlett-Packard LaserJet II. Dieser Drucker (oder vielmehr seine Kommandosprache) hat sich als Standard für Seitendrucker durchgesetzt. Der Betrieb eines HP-kompatiblen Druckers ist an moderner Software problemlos möglich. Darüber hinaus verfügen aber fast alle Drucker noch über andere Emulationen.
Kommen wir zur Vorstellung der Kandidaten.

Der Brother HL-4 wurde auf der CeBIT '90 zum ersten Mal der Öffentlichkeit gezeigt. Er basiert auf einer Vier-Seiten-Engine von Canon, die ebenfalls im nächsten Gerät, dem HP LaserJet IIp, ihren Dienst versieht. Um auf eines der Ergebnisse vorzugreifen: Diese beiden Maschinen sind alles andere als identisch.

Eine Neuheit der Orgatech '90 war NECs Vorstellung einer neuen Laserdrucker-Familie. Der NEC Silentwriter2 S60 ist deren kleinstes Mitglied. Die drei weiteren Prüflinge sind keine solchen Newcomer. Sie sind teilweise seit Jahren bewährt am Markt vertreten. Es sind dies der OkiLaser 400, der kleinste von OKI, der Sanyo SPX-608 und der Seikosha OP-105.

Umgewöhnung

Für den mit Nadeldruckern Vertrauten bringt der Umstieg auf den Laser- oder LED-Drucker erst einmal eine Reihe von Veränderungen mit sich. Allein das räumliche Problem. Die Geräte sind groß, oft über 15 kg schwer und verlangen Zugänglichkeit von mehreren Seiten. Neben Rechner und Festplatte steht nun ein weiterer Püsterich im Raum, denn alle Drucker werden per Lüfter gekühlt.

Dann das Papier. Billiges Endlosmaterial ist out - Einzelblatt ist in. Das kann zwar auch umweltschonend holziges sein. Doch bereits bei der Wiederverwendung der Rückseiten, die die Sparsamkeit gebietet, streikt so mancher Einzug. Viele werden ihre Arbeitsweise ändern, denn schnelles Tippen und Drucken einiger Zeilen bestraft der neue Drucker mit dem Verbrauch eines ganzen A4-Blattes, wie es sich für einen 'Seitendrucker' gehört. Das Mitlesen beim Druck, das den Vorgang beim Nadeldrucker erträglich machte - es fällt jetzt aus. Die Erstellung einer Seite vollzieht sich

vollständig im Innern der Geräte.

Neben solchen Nachteilen bieten die Seitendrucker aber so viele Annehmlichkeiten, daß man sich bald fragt, wie's eigentlich ohne ging. Und der schnell verstaubte Nadeldrucker wird allenfalls noch bemüht, wenn es um Durchschläge geht. In erster Linie ist die Schriftqualität begeisternd. In Verbindung mit entsprechender Software, die z.B. mit eigenen Pixel-Fonts arbeitet, sind die Druckergebnisse immer wieder eine Augenweide.

Das gilt ebenfalls für die Grafik. Vorbei sind die Zeiten, da die schönsten Elaborate durch die für Nadeldrucker so typischen Steifen verunstaltet wurden. Wer mit CAD oder Platinen-Layout arbeitet, wird seine Freude an den präzisen Linien und scharfen Konturen haben. Wer - wie der Autor - in einem hellhörigen Haus lebt, kann nun auch des Nachts arbeiten - Seitendrucker sind leise Vertreter ihrer Art.

Und - last not least - dem Programmierer bietet die Steuersprache 'PCL', die alle HP-Kompatiblen verstehen, ein völlig neues Betätigungsfeld. Besonders die Möglichkeit, die gesamte Seite anzusprechen, und die damit so einfache Grafikausgabe stimuliert algorithmisch denkende Geister.

Blick ins Innere

Die Technik der Seitendrucker

Der Weg, auf dem Seitendrucker Texte und Bilder zu Papier bringen, heißt korrekt 'photoelektrischer Prozeß'. Der hat tatsächlich Ähnlichkeiten mit dem photographischen Prozeß, beruht aber auf elektrostatischen Effekten. Im wesentlichen entspricht der Vorgang dem, der sich in jedem Photokopierer abspielt.

In der Prinzipzeichnung sehen Sie alle Bestandteile im Innern der Drucker, die zur Herstellung einer Seite beitragen. Im Mittelpunkt die Belichtungstrommel, eine aus Aluminium hochpräzise gefertigte und mit Photoleitermaterial beschichtete Walze. Sie dient als Überträgerin des Bildes auf seinem Weg von der Belichtung zum Papier. Der Kreislauf beginnt dabei an dem Ladedraht links oberhalb der Reinigungsstation.

Hier wird die Trommel gleichmäßig negativ geladen. Am Draht liegt eine Spannung von ca. 6000 Volt an; das durch ihn erzeugte elektrische Feld verschafft der Trommeloberfläche ein Potential von ca. -700 Volt. Im Bild ist das durch kleine negative Ladungen angedeutet. Im nächsten Schritt kommt die Belichtungseinheit zum Zug. Das kann ein Laserstrahl sein, der, von einem rotierenden Spiegel abgelenkt, die Walze überstreicht. Eine andere Möglichkeit ist, eine Leiste mit 2400 LEDs zu bestücken, die jeweils einen Abstand von einem 300stel Zoll besitzen. Die LCS-Drucker wiederum belichten mit einer Halogenlampe durch ein Flüssigkristallfeld, in dem einzelne Punkte undurchsichtig gemacht werden können. Die Unterschiede beschränken sich aber nur auf die Art, das Licht zu erzeugen, im übrigen sind die Prozesse identisch.

Durch das Licht, das auf die Trommel fällt, wird diese stellenweise elektrisch leitfähig. Dadurch findet ein Ladungsaustausch statt, in dessen Folge die Oberfläche fast vollständig an negativem Potential verliert. In diesem Stadium ist das Bild latent, d.h. es existiert zwar, doch ist es unsichtbar.

Im nächsten Schritt bringt die Entwicklungseinheit den Toner an die Trommel. Der Toner ist ein sehr feines, eisenhaltiges (und natürlich schwarzes) Pulver und ebenfalls negativ geladen. Er wird über eine Rolle, die nah an der Bildtrommel vorbeiläuft, auf diese übertragen. Genau die Stellen, die durchs Licht entladen wurden, ziehen den Toner buchstäblich herüber. Alle anderen stoßen ihn ab und werden nicht schwarz.

Nun existiert tatsächlich schon ein sichtbares Bild. Allerdings auf der Trommel und nicht auf dem Papier. Diesem Mißstand verschafft die Übertragungsstation Abhilfe. Wieder erzeugt ein Ladungsdraht, an dem ca. 5000 Volt anliegen, ein elektrisches Feld, das den Toner aufs Papier saugt. Von nun an trennen sich die Wege von Bild und Trommel. Letztere wird in der Reinigungsstation von überschüssigem Toner befreit, um wiederum negativ geladen zu werden etc.

Der Toner liegt aber noch lose auf dem Papier, erst die Fixierung schafft ein dauerhaftes Druckergnis. Dabei wird der Toner unter dem Druck zweier Walzen und einer Temperatur von ca. 150° Celsius ins Papier eingebrannt. Daher sind die Blätter, die den Drucker verlassen, häufig wellig.

Spätestens jetzt ist klar, warum die elektrostatischen Drucker seitenorientiert arbeiten, der Druckprozeß sich nicht abbrechen läßt. Wenn er auch in winzigen Schritten von einem 300stel Zoll abläuft, so können die Ladungseffekte nur zu einem sauberen Ergebnis führen, wenn sie kontinuierlich stattfinden. Welch wichtige Rolle dem internen Rechner zukommt, ist ebenfalls leicht zu ersehen. Nicht nur die Annahme der Daten aus dem Rechner, sondern auch die Erstellung der Bit-Matrix, die der Lichtstrahl abbildet, obliegt ihm.

Dieses Bit-Muster muß bereits vor dem Druck für die gesamte Seite aufgebaut und gespeichert werden. Wenn - wie bei Bildern - tatsächlich jeder Fleck einer A4-Seite bedruckt werden soll, müssen

8 Zoll (Breite) * 300 DPI * 11,5 Zoll (Höhe)
* 300 DPI = 8,28 Millionen Bits

zur Verfügung stehen. Das sind ca. 1000 kByte. Ein solcher Speicherausbau ist demnach für DTP- und Grafikanwendungen obligatorisch.

Bastelstunde

Sprach da jemand von 'plug and play'? Jedenfalls nicht beim heutigen Testfeld. Vor den Druck mit LED- und Laserdrucker haben die Verpackungsdesigner eine Orgie in Styropor und die mehr oder minder umfangreiche Montage gesetzt. Da die Belichtungseinheiten bei den vier Geräten von Brother, Hewlett-Packard (kurz: HP), NEC und Seikosha zu einem Cartridge zusammengefaßt sind, gestaltet sich deren Zusammenbau einfach. Die Handbücher weisen dem Neuling die nötigen Griffe und so ist der Drucker schnell startbereit. Einzig das simple und schlecht illustrierte Manual des Seikosha hinterließ ein schlechtes Gefühl. Aber auch der Zusammenbau des Okis und des Sanyos sind keine hoffnungslosen Unterfangen. Trotzdem sollten sich die Autoren der Anleitungen mal eine Scheibe bei HP abschneiden. Was die Erklärung notwendiger Handgriffe betrifft, sind die Amerikaner unerreicht.

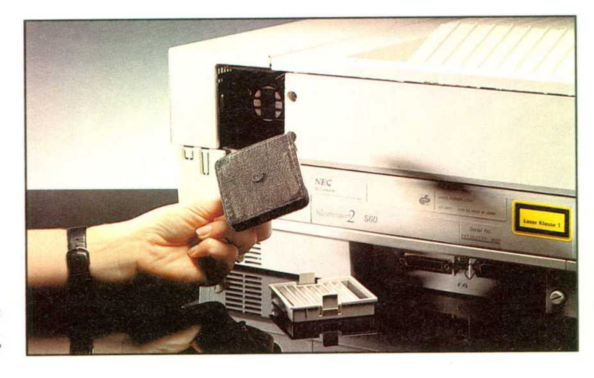
Im Wesentlichen sind die sechs Maschinen ähnlich aufgebaut - mit Ausnahme des Sanyo. Er kann seine Verwandtschaft mit Photokopierern nicht leugnen. Sein Innenleben ist von der linken Seite her zugänglich. Dort findet man die Bestandteile des Druckwerks (Tonerpatrone, Entwicklungseinheit, Fixierstation) zum Herausziehen. Die Wartung des Geräts ist nicht schwieriger als bei anderen, allerdings blieben dabei die Finger nicht immer sauber.

Bei den anderen Maschinen verbirgt sich das Druckwerk unter einer Klappe in der Ober - beim HP und beim Brother in der Schmalseite. Der Oki, NEC und Seikosha sind geradezu klassisch aufgebaut, die einzelnen Komponenten sind im Innern deutlich getrennt. Brothers HL-4 und HPs IIp hingegen sind wahre Meisterwerke der Komprimierung. Ihre Druckwerke nehmen kaum die Hälfte des Gehäusevolumens in Anspruch.

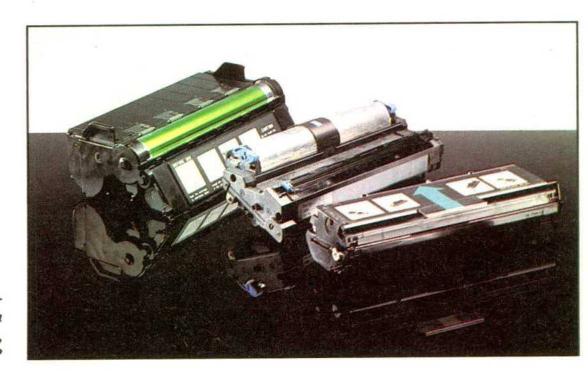
Unterschiede gibt es ebenfalls in den Wegen des Papiers durch die Maschinen. Bedingt durch die Lage von Papierzuführung und -ablage ergeben sich unterschiedliche Führungen durch die Druckergehäuse. Daraus entstehen typbedingt Probleme bei speziellen Papiersorten. Optimal ist ein Lauf, der möglichst nicht gekrümmt ist. Betrachten Sie die Abbildung, die ihn beim HP/Brother-Druckwerk zeigt. Das



Bastelstunde: Zum Einbau von Emulations- und RAM-Karten braucht man oft Fingerspitzengefühl.



Ozonfilter halten das Gas zurück. Hier beim NEC



Die Entwicklungseinheiten im Vergleich: Seikosha, Oki, Brother/HP

Papier wird - aufsummiert - um fast 250 Grad gekrümmt. Da es sowohl hin- wie zurückgebogen wird, hält sich die Welligkeit in Grenzen. Dagegen mutet der Seikosha seinem Material nicht einmal 90 Grad zu. Beim Oki und beim NEC beträgt dieser Winkel nur 180 Grad, beim Sanyo wiederum fast 360.

Bis auf den Seikosha lassen sich aber bei allen Geräten die Papierwege verkürzen, indem man Klappen öffnet. So werfen der Brother und der HP das Blatt dann vorn, nicht oben aus, ebenso der Sanyo. Der Oki und der NEC ähneln sich im Aufbau auch hier: Ihre Face-Up-Ablagen befinden sich an der Rückseite. Der genannte Winkel reduziert sich so auf Null. Auf diesen Ablagen liegen die Druckerzeugnisse mit dem Gesicht nach oben, also falsch herum sortiert.

Druckmedien

Die Hersteller der Seitendrucker machen in den Handbüchern immer wieder strenge Vorschriften, was die Wahl des Papiers betrifft. Durch den elektrostatischen Druckprozeß allein muß es Anforderungen bezüglich der Stärke, der Leitfähigkeit etc. erfüllen. Am besten geeignet ist normales Kopierpapier mit einem Quadratmetergewicht von 80 Gramm. Das darf auch ungebleichtes sein, doch sollte man ein wenig experimentieren. Denn nicht jeder Toner brennt sich in jede Papiersorte gleich gut ein.

Exotische Druckmedien wie Folien oder Aufkleber, aber auch Umschläge, unterliegen noch strengeren Anforderungen. Die Folie darf auf keinen Fall in der Fixiereinheit schmelzen, der Drucker könnte beschädigt werden. Das gleiche gilt für die Klebstoffe der Aufkleber. Man sollte tunlichst die speziell für Kopierer und entsprechende Drucker angebotenen benutzen. Sie sind zwar sündhaft teuer, aber wenn's die Maschine mit langem Leben lohnt...

Fast alle Geräte druckten klaglos auch auf festen Zeichenkarton. Mit Ausnahme des NEC. Obwohl er eine fast krümmungslose Papierführung bietet, macht er sich den Vorteil nicht zunutze und zog das Blatt erst gar nicht ein. Der Oki wie der Sanyo und der Seikosha hatten dabei die Nase vorn. Bei HP und Brother muß man ein wenig schieben, ehe sie den Karton annehmen.

Normales Papier ziehen die Kandidaten aus Magazinen ein, das beim Brother und beim HP lediglich aus einer simplen Klappe besteht und so geringe Kapazität bietet. Hier entfällt die Notwendigkeit einer manuellen Einzelblattzuführung. Eine solche erleichtert es ungemein, zwischendurch mal einen Umschlag zu drukken, ohne dabei großartig umbauen zu müssen. Geradezu vorbildlich ist diese Option beim Oki und beim NEC realisiert: Die Geräte merken von selbst, wenn ein Blatt am manuellen Einzug anliegt und ziehen es dann ein. Dem Seikosha und dem Sanyo muß man mittels Bedienfeld zuvor mitteilen, daß man den Handeinzug wünscht - ein unnötig komplizierendes Unterfangen.

Welcome To The Machine

Alle sechs heutigen Kandidaten kommunizieren über LCD-Displays und unterschiedliche Mengen an LEDs mit dem Benutzer. Über das Display werden Grundeinstellungen wie die Wahl der Emulation, des Fonts, des Zeichensatzes etc. getroffen. Diese Auswahl ist immer in Form von hierarchischen Menüs angelegt, durch die man sich mit den Tasten hangeln muß. In den Handbüchern finden sich umfangreiche Erklärungen zu diesen Menüs, so daß deren Handhabung bei keinem Gerät ein ernsthaftes Problem darstellt.

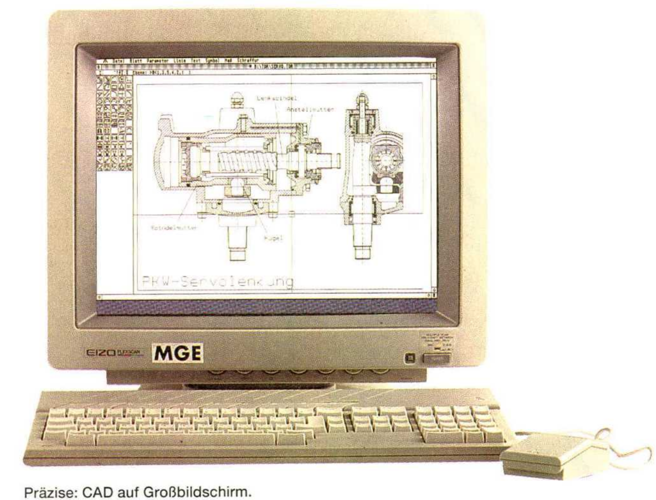
Aber auch Statusmeldungen wie 'Papierstau', 'OnLine' oder 'Aufwärmen' (während des Aufheizens der Fixiereinheit) werden über die Displays ausgegeben. Den Vogel schießt der Sanyo SPX-608 mit seiner 'Good Day'-Meldung nach dem Einschalten ab. Wer seine Werke bisher via Nadeldrucker ausgab, der ist gewohnt, bei einem Fehldruck einfach den Drucker aus- und einzuschalten, um ihn am Weiterdrucken zu hindern. Bei Seitendruckern ist das nicht möglich. Die Maschine während des Druckes auszuschalten, würde bedeuten, das Blatt hinterher herausfummeln zu müssen. Stattdessen schaltet man OffLine, wartet brav, bis das Gerät die laufende Seite ausgespuckt hat, und führt dann einen Reset des Drucker-

Um Ihre Anwendungen brauchen Sie sich bei unserer MGE-Grafikkarte keine Sorgen machen, denn Sie arbeitet automatisch mit ihnen zusammen. Egal, ob Sie die 16,7 Millionen Farben der MGE ins Spiel bringen oder lieber mit 256 Graustufen arbeiten wollen, die MGE zeichnet sich durch ihre Flexibilität aus. Durch variable Auflösungen von bis zu 1664*1200 Bildpunkten läßt sich jeder Monitor optimal nutzen. Ein leistungsfähiger, hochmoderner Grafikprozessor läßt den Bildaufbau mit der MGE in atemberaubender Geschwindigkeit vonstatten gehen. Dies sind die Merkmale, die die MGE zur idealen Grafikerweiterung für jede Anwendung macht. Auf dieser Doppelseite wollen wir Ihnen ein paar Anwen-



Sinnvoll: Color-DTP mit CALAMUS SL

Anwendung



Präzise: CAD auf Großbildschirm.

Besonderheiten: Anschlußmöglichkeit für Genlock oder externe Synchronisation, Sockel für mathematischen Coprozessor 68881

Software: Installationsprogramm, Treiber für Hardware, GDOS-VDI und Drucker, Monitorbibliotheken, Kontrollfeld als Accessory, diverse Demos und Bilder, Programmierbibliotheken.

Bildspeicher: 1024 kByte
Farbpalette: 16,7 Millionen
gleichzeitig darstellbare Farben/Graustufen: 256
Pixel-Frequenzen: 13.75, 27.5, 55 und 110 MHz

Betriebsarten: Interlace/Non-Interlace
Monitoranschluß: ECL und RGB-analog

Typische Auflösungen:
Interlaced (Halbbildfrequenz):
1664 x 1200 Pixel monochrom, 88 Hz
1280 x 1024 Pixel in 16 Farben, 65 Hz
896 x 684 Pixel in 256 Farben, 66 Hz
Non-Interlaced (Vollbildfrequenz):
1280 x 960 Pixel monochrom, 65 Hz
896 x 688 Pixel, in 16 Farben, 67 Hz
640 x 480 Pixel in 256 Farben, 66 Hz

Preise und Systeme:

MGE II	DM 1998.-
MGE II mit Eizo 9070S (16"-Farbmonitor)	Preis auf Anfrage
MGE II mit Eizo 6500 (21"-Graustufengroßbildschirm)	Preis auf Anfrage
MGE II mit Eizo 9400 (20"-Farbgroßbildschirm)	Preis auf Anfrage



Brilliant: Farbbildbearbeitung mit CRANACH auf Großbildschirm.

dungen vorstellen, die natürlich nur ein paar Vorschläge zur Nutzung der MGE sein sollen. Speziell ausgerichtet auf die Anforderungen kreativer Anwendungen wie DTP, CAD oder elektronische Bildverarbeitung, eröffnet sie neue Dimensionen der Produktivität und verwandelt den Mega ST in eine Grafik-Workstation. Doch auch wenn Sie lieber eine Datenbank oder eine Textverarbeitung nutzen wollen, werden Sie schnell die Vorteile der MGE schätzen lernen. Die Treiber-Software sorgt für problemlosen Betrieb in allen Auflösungen, egal ob Sie das Einsteigermodell oder das Flaggschiff eines Monitorherstellers benutzen. Das MGE-Kontrollfeld ermöglicht jederzeit eine freie Einstellung von Auflösung und Farben.

anmelden!

Übersichtlich: SciGraph-Diagramme in Farbe auf Großbildschirm.

Gerne führen Ihnen die folgenden Händler die MGE vor: Ernst Brinkmann KG, Spitalerstr. 10, 2000 Hamburg 1, Tel. 040/30040 • **Sienknecht Bürokommunikation**, Heiligengeiststr. 20, 2120 Lüneburg, Tel. 04131/46122 • **MCC**, Holzkoppelweg 19a, 2300 Kiel 1, Tel. 0431/54381 • **PS-DATA, Hard- und Software GmbH**, Faulenstr. 48-52, 2800 Bremen, Tel. 0421/170577 • **Kurt Neumann PC**, Bürger 160, 2850 Bremerhaven, Tel. 0471/42006 • **COM-DATA GmbH**, Schiffgraben 19, 3000 Hannover 1, Tel. 0511/326736 • **3K Computerbild GmbH**, Sassenfeld 71, 4054 Nettetal 1, Tel. 02153/60001 • **CSA Computersysteme**, Hüttenstr. 56, 4650 Gelsenkirchen, Tel. 0209/203420 • **Data Pach GmbH & Co KG**, Neuhausstr. 4, 5100 Aachen, Tel. 0241/4778537 • **Werbestudio Abakus**, Römerstr. 24, 5300 Bonn 1, Tel.: 0228/635712 • **Lehr Bürocenter GmbH**, Güterstr. 82, 5500 Trier, Tel. 0651/209710 • **Eickmann Computer**, In der Römerstadt 249, 6000 Frankfurt 90, Tel. 069/763409 • **Hard- & Softwareservice**, Christine Veigl, Adalbertstr. 61-63, 6000 Frankfurt 90, Tel. 069/774043 • **Pauly Büromaschinen**, Salzgasse 6, 6250 Limburg/Lahn, Tel. 06431/50040 • **WAVE Computersysteme,** Südanlage 20, 6300 Gießen, Tel. 0641/72357 • **Schreiber Computer**, Rotebühlplatz 10, 7000 Stuttgart 1, Tel. 0711/221996 • **Walliser & Co KG**, Marktstr. 48, 7000 Stuttgart 50, Tel. 0711/567143 • **Comp & Phone**, Alleenstr. 66, 7312 Kirchheim-Teck, Tel. 07021/3949 • **GCS Röseler**, Jahnstr. 1, 7343 Kuchen, Tel. 07331/81925 • **Besch & Partner**, Stuttgarter Str. 53, 7432 Bad Urach, Tel. 07125/8199 • **Erhardt Bürotechnik**, Am Ludwigplatz, 7500 Karlsruhe 1, Tel. 0721/16080 • **Jöst Computer**, Im Wendelrot 5, 7520 Bruchsal, Tel. 07251/103091 • **Computer Ludwig City Studio**, Am Rindermarkt 6, 8000 München 2, Tel. 089/2609801 • **JOBI'S Unternehmenssupport DV & Marketing**, Fliegenstr. 12, 8000 München 2, Tel. 089/2608703 • **Computer Corner**, Landshuter Str. 4, 8313 Vilsbiburg, Tel. 08741/4211 • **HIB Computer DTP-Center**, Äußere Bayreuther 57a-59, 8500 Nürnberg 10, Tel. 0911/995140 • **CCN Computer Center, Nürnberg GmbH**, Gibitzenhofstr. 86, 8500 Nürnberg 70, Tel. 0911/421056 • **Sekui & Partner GmbH**, Karmelitenstr. 26, 8700 Würzburg, Tel. 0931/57555 • **V.Willgerodt Bürobedarfshaus GmbH & Co KG**, Hanauer Str. 12, 8750 Aschaffenburg, 06021/21375 • **Adolf & Schmoll Computer**, Schwalbenstr. 1, 8900 Augsburg, Tel. 0821/ 528533 • *Händleranfragen erwünscht!*

Machen Sie Ihre erste 16-Million!



Schwalbacher Str. 52 • 6236 Eschborn
Tel.: 0 61 96 / 4818 11 • FAX: 06196/41885

Groschengräber

'Immer, wenn es neben mir Klick macht, bin ich um einen Groschen ärmer.' Das sagt nicht etwa einer, der unter Spielautomatensucht leidet. Jeder, der einen Seitendrucker sein eigen nennt, gelangt zu dieser Erkenntnis.

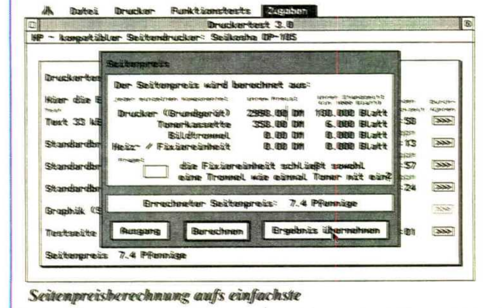
Ungeachtet der immer schneller sinkenden Kaufpreise für die Geräte selbst, sind für den Betrieb der Laser- oder LED-Drucker eine Reihe von Verbrauchsmaterialien erforderlich. Wer stolzen Griffs den Nadeldrucker unter den Tisch verbannte, um das neue, elektrostatische Gerät darauf zu stellen, wird sich umgewöhnen müssen. Ein Farbband konnte man zumindest so lange benutzen, wie es irgendwie Spuren hinterließ. Der Toner im neuen Drucker ist - wenn er verbraucht ist - richtig alle. Da hilft kein Ach und Weh: neuer muß her. Das riecht nach neuerlicher Geldausgabe. Flugs greift man zum Taschenrechner, um den finanziellen Aufwand je Seite zu berechnen, da taucht das erste Problem auf: Wielange hält eigentlich eine Toner-Füllung?

Der Blick ins Handbuch enthüllt: 2.000 bis 6.000 Seiten bei fünfprozentiger Schwärzung des Papiers sollen die Patronen (englisch: Cartridges) aushalten. Was bitte sind 5% Schwärzung? Ich habe wieder keine Mühen gescheut und für Sie die Bits gezählt. Wenn Sie den Norm-Geschäftsbrief ('Dr. Grauert') in einer 12 Punkt hohen Helvetica-Schrift drucken (was normaler Textausgabe entspricht), dann setzen Sie genau 4,7% aller möglichen Punkte auf dem DIN A4-Blatt. Nachtigall, ich hör' dir trapsen. Die Angabe aus dem Manual ist also schnell relativiert. Nur bei einem kurzen Text (1250 gedruckte Zeichen auf einer Seite sind nicht viel) gilt diese Angabe. Gescannte Bilder etwa können um das Vielfache mehr Toner verlangen. Gehen wir im weiteren trotzdem vom wortkargen Textdruck aus - es ist die einzige Angabe von Herstellerseite. Aber genießen Sie den Wert mit Vorsicht!

Je nach Aufbau der Maschine ist der Toner nicht die einzige Komponente, die sich verbraucht. Wenn er nicht in einer gemeinsamen Kartusche mit ihr untergebracht ist, gehört auch die Bildtrommel dazu. Für sie sind Standzeiten zwischen 10.000 und 20.000 Seiten typisch. Die hängt jedoch weniger von der Seitenzahl als von der Art der Nutzung ab. Wer oft auf Karton oder Umschläge druckt, wird nicht lange Freude an ihr haben, der Serienbriefdrucker darf seine mit Sicherheit länger behalten. Versuchen Sie jedoch, geklammertes Papier zu bedrucken, wird die Standzeit auf 1 Blatt sinken.

Zudem ist - meist alle halbe Jahre - ein neuer Ozonfilter fällig. Nun wird es schwierig, denn wieviel Blätter werden denn in diesem Zeitraum bedruckt? Auch hier müssen wir von Durchschnittsangaben ausgehen. Büromaschinen sind für eine Lebensdauer von fünf Jahren ausgelegt. Bei einer monatlichen Kapazität von 3000 Seiten (genau die haben alle Drucker in diesem Vergleich) kommen Sie also auf ein Gesamtvolumen von 180.000 Blatt. Der Filter ist also nach 18.000 Drucken zu ersetzen. Glücklicherweise sind die Filter bei fast allen Maschinen der Trommel beigelegt, verursachen also keine weiteren Kosten. Damit hätten wir die Standzeiten aller Komponenten. Die der Maschine ergibt sich aus der Kapazitätsangabe. Bei Druckwerken, die für größere Mengen konzipiert sind als unsere 'Kleinen' im Test, muß noch eine neue Fixiereinheit (Heizung und Walzen), manchmal eine komplett neue Mechanik, nach ca. 180.000 Seiten mitgerechnet werden. Das gilt z.B. für Ataris SLM 605 [1].

Die Rechnung für den Seitenpreis geht also so: Man berechne für alle Komponenten, wie oft sie in einem Druckerleben gebraucht werden. Das Ganze minus eins; denn wenn Sie das Gerät kaufen, ist es ja bereits komplett. Dann multipliziere man diese Werte mit den Preisen der jeweiligen Komponenten und teile die Summe aller Preise durch die Standzeit der Maschine.



Seitenpreisberechnung aufs einfachste

Wenn Ihnen das jetzt noch ein wenig spanisch vorkommt, können Sie sich auch das Druckertestprogramm hernehmen, dort den Menüpunkt *Seitenpreisformel* anwählen und im abgebildeten Dialog alles eintragen. Ihr Rechner übernimmt dann den Rest.

[1] Auf ein Neues - Test Atari SLM 605, ST-Computer 12/90 S.172ff.

speichers durch. Doch nicht alle Geräte halten sofort an, der Sanyo z.B. wollte erst seinen kompletten Speicherinhalt drucken. Da hilft nur, ihm die Papierkassette zu rauben, um ihn von seinem sinnlosen Tun abzuhalten...

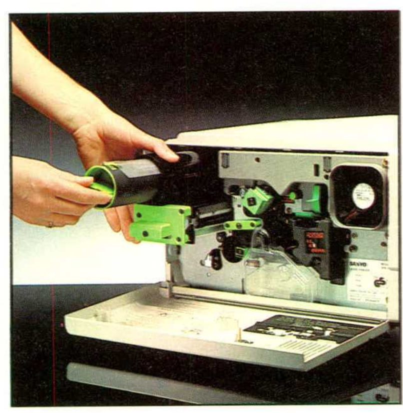
Die detaillierten Statusmeldungen, die oft auch auf die Quelle des Problems hinweisen ('Papierstau im Einzug'), erfordern eine sensible und aufwendige Elektronik. So überwachen eine Reihe von Tastern und Fühlern im Innern den Lauf des Papiers, den Zustand der einzelnen Druckerkomponenten etc. Oft wird es dadurch schwer, das Gerät zu zweckentfremden. Wenn z.B. dem NEC beigebracht wurde, er würde fortan mit DIN-A4-großen Blättern gefüttert, dann geht er davon aus, daß der Papierfühler am Einzug eine bestimmte Zeit lang heruntergedrückt bleibt. Zieht man nun einen Umschlag ein, hält der Drucker an und meldet einen Papierstau. Denn der zu kleine Umschlag gab den Fühler zu früh frei. Konsequenz: Über jeden Papierformatwechsel müssen die Drucker informiert werden.

Die Rechner, die im Innern der Seitendrucker ihre Bits schieben, hätten noch vor kurzer Zeit auch auf dem Tisch ein gutes Bild abgegeben. So findet sich im Seikosha und im Sanyo Motorolas 68000, ein volles Megabyte an Speicher ist zum Grafikdruck obligat. Neben der Aufbereitung der kompletten Seite und der Überwachung des Druckablaufs führen die internen Rechner auch eine Selbstdiagnose durch, die vielfach nur aus der Kontrolle eines Seitenzählers besteht. Meldungen wie 'Toner leer' sollte man tunlichst mit Skepsis betrachten. Wer weiß, wie sie zustandekommen.

Mit softwaretechnischen Finessen warten der Brother HL-4 und der Sanyo SPX-608 auf: Sie sind in der Lage, ankommende Grafikdaten komprimiert abzulegen, um so Speicher zu sparen. Der Gedanke ist naheliegend, denn gerade Grafikinformationen lassen oft hohe Komprimierungen zu. Allerdings kann eine solche Technik, die ja ohne Zeitverzögerung arbeiten soll, nicht dermaßen effektiv sein wie spezielle Programme, die den Datenbestand zunächst untersuchen können.

Leider zeigten sich dann auch schnell die Grenzen dieser Einrichtungen. Der Brother druckte ca. zwei Drittel des formatfüllenden Testbildes, bevor er mit einer Fehlermeldung abbrach und auf dem nächsten Blatt fortsetzte. Immerhin, fast 640 kB paßten in den 512 kB großen Standardspeicher. Ganzseitige Grafik gelang mit 'DACT' (=Data Compression-Technology) immer dann, wenn die Seite viel Weiß enthielt, offensichtlich ist der Drucker darauf ausgelegt - weniger darauf, sich wiederholende Muster zu erkennen. Nach dieser Erfahrung blieb uns allerdings nichts anderes übrig, als dem Brother trotz 'DACT' ein weiteres Megabyte an Speicher zu gönnen.

Enttäuschung dann beim Sanyo. Trotz der 1024 kB Speicher, mit denen er ausgeliefert wird, fanden nur knapp 700 kB des Testbildes



Der Sanyo zeigt einen ähnlichen Aufbau wie ein Kopierer: hier das Einsetzen des Toners

darin Platz. Von Kompression zeugte lediglich, daß Seiten entsprechender Datenmenge (rund 970 kB) dann vollständig gedruckt wurden, wenn ihr Weißanteil höher war als beim Testbild. Bei allen anderen Geräten wurde der Test mit Speicherkonfigurationen von 1 MB (NEC S60) oder 1,5 MB (HP LaserJet IIp, OkiLaser 400, Seikosha OP-105) durchgeführt.

Öko-Druck

Spätestens, wenn der erste Toner sein Ende durch leere Seiten kundtut und der Nachfüllsatz bereits eingesetzt ist, dann steht jedermann und jedefrau vor der Frage: wohin mit dem alten? Besitzer des OkiLaser 400 und Sanyo SPX-608 haben da noch die geringsten Probleme. Sie tauschen alle 2.500 bzw. 3.000 Blatt lediglich eine Röhre, die Tonerpatrone. Sie besteht beim Oki aus Aluminium, beim Sanyo vorbildlich aus Pappe.

Besitzer der vier anderen Maschinen bezahlen die simplere Wartung ihrer Geräte mit einem erheblichen Mehr an Müll. Interessanterweise (oder leider?) macht sich das nicht in den Kosten einer Seite bemerkbar. (Lesen Sie dazu auch den Extratext 'Groschengräber'). In ihren Entwicklungs-Cartridges sind der Tonerbehälter, Toner, Bildtrommel, zum Teil auch die Ladungsdrähte sowie die notwendige Mechanik zusammengefaßt. Das alles wandert nach 3.500 bis 6.000 Blatt auf den Müll. Bei höherem Schwarzanteil als 5%, der etwas gering angenommen ist, sogar noch früher.

Auf den Müll? Es blutet dem Sparsamen das Herz, und der Tüftler beginnt, Nachfüllmethoden zu ersinnen. Für die gerade in den USA sehr verbreiteten HP-Geräte gibt es auch hierzulande Firmen, die einen Nachfüllservice anbieten. Verständlich sind die Zweifel, die Hersteller wie HP dagegen hegen. Die Cartridges sind nur für eine gewisse Lebensdauer ausgelegt, was ihren relativ geringen Preis erklärt. Wer sie ein öfter benutzt, läuft Gefahr, an Qualität einzubüßen. Und das fiele letztlich auf den Hersteller zurück.

Trotzdem haben diese Aktivitäten von Fremdfirmen durchaus ihr Positives gehabt. Denn die wahren Kosten des Müllberges trägt nicht der Hersteller, nicht der Kunde, sondern die Allgemeinheit. So haben die Hersteller, die sich zusätzlich unter dem wachsenden Druck der Öffentlichkeit sehen, darüber nachdenken müssen, ein eigenes Recycling-Konzept anzubieten. Mittlerweile scheint sich auch hierzulande ein Konsenz in Richtung Müllvermeidung durchzusetzen, und flugs hat - allen voran - Hewlett-Packard sein in den USA längst etabliertes Rückführungsprogramm nach Deutschland exportiert.

Die Bad Homburger stehen gut da: Sie sind die ersten, die ein funktionales Rückführungsnetz dieser Größenordnung vorzeigen können. Nicht zu vergessen ist, daß es gerade deren Maschinen sind, die ein solches Konzept erst notwendig machen. Es ist damit das Symptom bekämpft - die Müllvermeidung beginnt viel früher, nämlich in Konstruktion und Herstellung. Wir haben alle Hersteller dieses Vergleichs zum Thema Rücknahme gefragt. Hier die Antworten:

- Brother: Cartridge wird zurückgenommen, zerlegt und eingeschmolzen.
- Hewlett-Packard: Cartridge wird zurückgenommen, zerlegt und eingeschmolzen.
- NEC: Cartridge wird nicht zurückgenommen, kommt in den Hausmüll.
- Oki: Tonerbehälter wie Bildtrommel werden nicht zurückgenommen, kommen in den Hausmüll.
- Sanyo: Tonerbehälter wie Bildtrommel werden nicht zurückgenommen, kommen in den Hausmüll.
- Seikosha: Cartridge wird nicht zurückgenommen, kommt in den Hausmüll.

Ein trübes Bild also, einzig Oki kündigt für die nahe Zukunft die Vorstellung eines Recycling-Programms an.

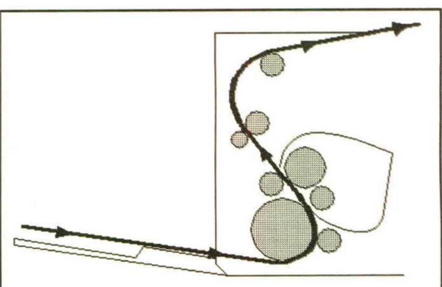
Aber der Druck mit Laserstrahl oder LED-Licht belastet die Umwelt noch auf andere Weise. Neben den Tonerresten, die während des Druckvorganges bei der Reinigung der Bildtrommel anfallen, entstehen vor allem Ozon und Lärm.

Die Tonerreste werden nur beim Sanyo nicht wieder in den Tonerbehälter zurückgebracht. Er hat einen kleinen Behälter, der mit der Tonerpatrone weggeworfen wird - schade drum. Brother behauptet, daß der HL-4 keinen Ozonfilter brauche, da er praktisch keines erzeuge. (Ozon ist dreiwertiger Sauerstoff, der durch die hohen elektrischen Felder aus dem Luftsauerstoff entsteht. Es baut sich zwar schnell ab, ist aber schädlich für die Lunge.) HP hingegen baut in den IIp, der ja auf der gleichen Maschine basiert, einen Ozonfilter ein. Genau diese Maßnahme ergreifen auch alle anderen Produzenten.

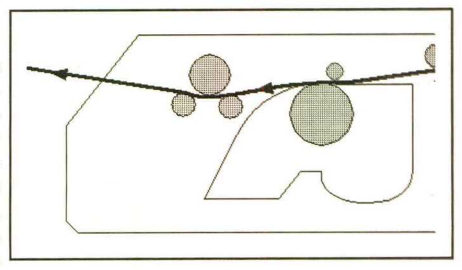
Bleibt der - offensichtlich - unvermeidliche Lärm des Bürogerätes Seitendrucker. Die Maschinen werden recht warm, die Heizung immerhin 150°. Da ist ein Lüfter unvermeidlich. Doch warum er so unverschämt laut sein muß wie beim NEC, bleibt ungeklärt. Das Brother/HP-Druckwerk verhält sich weit nervenschonender: Es senkt die Lüftung und die Heizung auf ein Minimum ab, wenn der Drucker keine Daten empfängt. Ebenso pfiffig ist der Oki. Bei ihm kann man die 'Time to quiet' sogar einstellen, denn das immerwährende Aufheizen erhöht die Wartezeit für die erste Seite.

Die Tests

Nach diesem Exkurs durch die Thematik der Seitendrucker soll nicht im Dunkeln bleiben, welche Tests wir durchgeführt haben, um Geschwindigkeit und Handling der Geräte zu ermitteln. Dem Interessierten sei empfohlen, sich im nächsten Monat (erst ab der Juni-Ausgabe) die entsprechenden MAXON-PD-Disketten zu besorgen, denn es gibt ein neues Druckertestprogramm! Neben großen Erweiterungen im 24-Nadelbereich haben nun auch dort die Seitendrucker Einzug gehalten. Momentan ist es zwar noch auf die HP-Kompatiblen beschränkt und ignoriert die Steuerkommandos von Fujitsu, Canon, Epson und Kyocera, doch dürfte damit ein wesentlicher Teil abgedeckt sein. Aus dem neuen Programm (das während



Papierweg beim Brother HL-4 und beim HP IIp



Papierweg beim Seikosha OP-105

des Testes noch von Fehlern befreit und erweitert wurde und daher erst in 4 Wochen für die PD fertig sein wird) stammen sämtliche Zeitmessungen, die wir mit den sechs Kandidaten durchgeführt haben.

In den nachfolgenden Einzeltexten lesen Sie jeweils, was uns zu den einzelnen Maschinen besonders auffiel. Sie finden dort eine Druckprobe, die in Originalgröße einen Ausschnitt aus unserem Testbild zeigt, sowie eine ST-Hardcopy mit bekannten Füllmustern. Diese Abbildungen sind sehr gut geeignet, die Grafikqualität zu beurteilen. Achten Sie dabei besonders auf die helle Stelle auf dem Ball, zu deren Wiedergabe der Drucker Grauwerte gut nuancieren muß.

Anhand des im Testprogramm enthaltenen Font-Tests ermittelten wir die druckerinternen Fonts und beurteilten ihre Qualität. Für den Test 'Text mit Pixel-Fonts' haben wir den Times10-Pixel-Font von Tempus Word genommen und mit jedem Drucker eine Seite in dieser Schrift ausgegeben. Die wichtigste Grundlage zur Kritik bildet die 'PCL-Testseite'. Sie besteht aus Text und Grafik, zeigt Lineale mit Zeilen- und Zollteilung sowie Füllmuster, wie sie die Steuersprache PCL selbst kennt. Dazwischen enthält sie jede Menge absolute und relative Positionierungen, so daß der Rechner im Drucker etwas zu tun bekommt. Sie entlarvte dann auch den NEC und ironischerweise den HP als nicht hundertprozentig kompatibel zum Vorbild, dem HP LaserJet II.

Für den Druck auf Folie benutzten wir lasertaugliche Overhead-Folie. Leider mit geringem Erfolg. Es zeigte sich, daß lediglich der NEC, der Oki und der Sanyo in der Lage sind, den Toner ausreichend in die Folie einzubrennen. Und auch dann ist er längst nicht abriebfest, und das Schwarz ist ungleichmäßig. Bei allen anderen verschmierte der Druck stark. Anders das Ergebnis auf laserfesten Aufklebern: Mit keinem Gerät ein Problem. Auch durch die HP/Brother-Engine mit ihrer hohen Papierkrümmung gingen sie glatt durch. Die jedoch hat ihre Probleme mit Umschlägen. Die stark verknitterten Ergebnisse sind nicht zum Verschicken geeignet. Auch der Seikosha mag sich mit ihnen nicht anfreunden: sie blieben in der Fixierstation hängen.

Zu der Tabelle mit den Geschwindigkeiten: Für die Spalte 'Text' wird der gleiche 15seitige ASCII-Text gedruckt wie auch bei den Nadeldruckern. Die Werte sind also vergleichbar. Der 'Brief' entpricht dem aus der DIN 32751 ('Dr.Grauert'). Er wird zunächst einzeln gedruckt, dann per Software-Befehl mit 10 Kopien. Dabei zeigt sich deutlich der Einfluß der Wartezeit bis zur ersten Seite, die bei fast allen Druckern erheblich ist. Interessant auch, daß lediglich der Seikosha OP-105 die vom Hersteller angegebene Geschwindigkeit von fünf Seiten pro Minute (Pages per Minute = ppm) in der Praxis erreicht.

Freiheit der Schrift

Seitendrucker und das Font-Problem

Von Nadeldruckern ist man gewöhnt, die gebotene Schrift auf vielfältigste Weise zu verändern. Stilistischen Ausschweifungen setzt der Drucker keine Grenzen: Kursiv, fett, hoch- und tiefgestellt, ja sogar Schattierungen und Umrißdruck sind kein Problem. Bei Seitendruckern sieht das ganz anders aus.
Die Fonts liegen den Druckern - gespeichert in ROMs - als Bitmap vor. Auf diese Bitmap lassen sich, ähnlich wie bei einem Malprogramm, verschiedene Effekte anwenden. Gerade die Verdopplung oder Halbierung von Höhe und Breite sind einfach zu bewerkstelligen. Etwaige Ungenauigkeiten, wie sie dabei zwangsläufig auftreten (wenn z.B. kursive Schrift errechnet wird), fängt ein Nadeldrucker durch die Unschärfe seines Druckes mühelos ab. Bei der Einführung von Seitendruckern mit der typischen Auflösung von 300 Punkten pro Zoll (das sind 0,085 Millimeter Abstand) tauchte aber das Problem auf, daß diese Treppchen und Zacken unbarmherzig abgebildet wurden. Das Druckbild forderte Konsequenzen.
Man begann, den Druckern jeden Stil als neuen Font beizugeben, in dem solche Schmankerln wegretouchiert sind. Wer einen HP LaserJet II sein eigen nennt, der kann gerade den der Schreibmaschine nachempfundenen 'Courier' mit dem Stil 'fett' versehen. Als zweiter Font ist noch der 'LinePrinter' drin, der nicht 10 Zeichen, sondern 16,6 Zeichen auf ein Zoll bringt. Bei ihm ist jeglicher Stil unmöglich.
Bald begann ein beispielloser Run auf Fonts jeglicher Machart. Hier sehen die HP-Clones ihre Domäne. Sie bieten oft bessere Font-Ausstattungen als ihr Vorbild - nicht selten bei niedrigerem Preis. Es bestehen jedoch drei weitere Möglichkeiten, Schriften in den Drucker zu bekommen.

1. Download. Unter diesem Vorgang versteht man die Übermittlung eines kompletten Fonts vom Rechner in den Druckerspeicher. Die sogenannten Soft-Fonts kann man kaufen oder sich aus PD-Sammlungen besorgen. Nachteile: Das kostet Druckerspeicher. Für einen 10 Punkt-Times sind etwa 20 kB zu veranschlagen. Außerdem muß der Vorgang bei jedem Druckerstart wiederholt werden. Darüber hinaus muß der Treiber der Textverarbeitung über die Schrift informiert werden.
2. Font-Karten. Alle Drucker verfügen über einen oder mehrere Einschubplätze für Font-Karten. Diese bieten je nach Hersteller unterschiedliche Mengen an Fonts. Sie sind oft recht teuer, fressen aber den Speicher nicht auf. Jedoch muß auch hier der Treiber in Kenntnis gesetzt werden.
3. Softwareeigene Pixel-Fonts. Sie verlangen höheren Aufwand an Speicher und Rechenaufwand auf seiten des Computers. Es entstehen praktisch keine Treiberprobleme, da die Daten grafisch übermittelt werden. Nachteil: Die Datenmenge für einfache Texte steigt leicht auf das Zehnfache.

Gerade im Bereich der IBM-kompatiblen Rechner haben sich die ersten beiden Methoden durchgesetzt. Große Textverarbeitungen wie WordPerfect arbeiten mit eigenen Download-Fonts, für die ihre Treiber angepaßt sind. Ein Teilproblem ist damit elegant umgangen. Aber auch PostScript hat auf dieser Situation sein Süppchen kochen können. Es arbeitet mit Vektor-Fonts, die hemmungslos verändert werden können. Einfache Textausgaben verlangen kaum mehr, als die reinen ASCII-Daten zum Drucker zu schicken. Die Rechenleistung steckt dann im Drucker. Denn jede PostScript-Ausgabe ist ein Programm, das durch einen Interpreter geschickt wird. Entsprechend kosten PostScript-Karten als Einschübe für HP-Laser nicht unter 1000 DM.

Die drei weiteren Tests decken den grafischen Teil ab. Hier werden die Daten nicht - wie vor - über die normale BIOS-Funktion zum Drucker geschickt. Auch das ist neu am Druckertestprogramm der Version 3.0: Eine Interrupt-Routine sorgt für sehr schnelle Datenübertragung zum Drucker und läßt gewiß keine Maschine warten. Die Übertragungsrate steigt so von ca. 2000 Bytes pro Sekunde (BIOS) auf maximal 16800 Bytes/s beim NEC S60!

Zunächst ist wieder einmal der DIN-Brief an der Reihe, allerdings jetzt gedruckt mit einem Pixel-Font der Auflösung 300 DPI. Dieser Test entspricht der Ausgabe mit Signum!. Hier vermindern viele Positionierungen die zu übertragende Datenmenge auf 172 kBytes für eine Seite. Das Testbild - mehrfach erwähnt - ist eine komplette DIN-A4-Seite in 300-DPI-Auflösung. Ein Schwarzweißfoto (vielen Dank an Bernd Opitz, den Photographen) wurde mit Retouche Professional gescannt und mit der Rasterweite 20,87 Linien/ cm gerastert. Satte 993022 Bytes ist es groß. Besitzer von weniger als 2 MB und Festplatte werden darauf verzichten müssen.

Die PCL-Testseite wurde schon erwähnt, bleibt die letzte Spalte, die Überraschendes offenbart. Das Testprogramm bietet den Gerä-

ten die Daten in einer Geschwindigkeit an, die auf einem Acht-Megahertz-Atari nicht mehr zu steigern ist. Doch was machen die Drucker daraus? Dafür haben wir die reinen Übertragungszeiten für das Testbild gesondert ermittelt und die Datenraten errechnet. Der Oki mochte gar nicht erst anfangen, die anderen Drucker verzögern die Abnahme bis auf Werte, die im Bereich der BIOS-Geschwindigkeit liegen. Einzig der NEC fällt hier positiv auf.

Ein großer Teil der Tests beschäftigt sich mit der Geschwindigkeit und der Qualität der Drukker bei der Erstellung von Grafiken. Dabei ist eine Beobachtung zu machen, die ebenso für Nadeldrucker gilt. Die meisten Drucker sind konsequent für Texte, und wenn überhaupt, dann nur für kleine Grafiken ausgelegt. Ganz so, wie es dem Anwendungsbereich IBM-kompatibler Rechner entspricht. Denn diese stellen für einen Druckerproduzenten die Hauptabnehmer dar.

Mit unseren Rechnern jedoch sind von jeher Bilder, auch größeren Formats, kein Problem gewesen. Geradezu logisch und ohne systemeigene Mängel aufzudekken, haben DTP und Ganzseitengrafik Einzug gehalten. Entsprechend anders sind hier die Anforderungen an einen Drucker. Wer mit Retouche Professional ein gescanntes Photo rastert, um es in eine Dokumentation einzusetzen, der wird es auch drucken wollen. Und genau der beginnt, seinen Drucker mit Datenmengen zu konfrontieren, für die - das hat der Test gezeigt - nicht alle Geräte gleichermaßen geeignet sind. Darüber hinaus stellt gerade ein solches Bild hohe Anforderungen an die Präzision eines Druckwerkes. Wer übrigens meint, mit Hilfe des Tonerreglers, mit dem sich bei allen Maschinen die Menge des aufgebrachten Toners regulieren läßt, die Druckqualität beeinflussen zu können, der irrt. Diese Regler sind nur dann sinnvoll eingesetzt, wenn sich der Toner dem Ende zuneigt. Sattere Schwärzen erkauft man sich sonst immer mit zulaufenden Mustern.

Und damit läßt sich der Kreis schließen zum Atari-Laser, der ja zu Beginn bereits angesprochen wurde. Oft hört man das Argument, 'normale' Seitendrucker hätten am Atari keine Chance, sie seien zu langsam. Es gibt nun mindestens ein Gerät, das eine Ganzseitengrafik mit 300 DPI, über Centronics ausgegeben, innerhalb nicht einmal eineinhalb Minuten zu Papier bringt. Und eine solche simple Ausgabe ist zeitfressend (leider benutzt Calamus sie). Z.B. Signum! oder Tempus Word arbeiten - wenn möglich - mit Positionierungen. So kann schon vom Treiber her die Datenmenge enorm verringert werden. Verbunden mit einem Drucker wie dem NEC, der die Leistung der Schnittstelle voll nutzt, bekommt der Atari-Laser durchaus Konkurrenz. →

Drucker Einzel	Text einzeln	Brief 10 Kopien	Brief Brief 300	Grafik S/W-Bild	Grafik Testseite	PCL	Seitenpreis	Datenrate an DRU_TEST.PRG
Brother HL-4	00:19 / 04:03 = 3.7 ppm	00:34 = 1.8 ppm	02:49 = 3.6 ppm	01:06	03:02	00:22 / 00:57	8.1 Pf.	6737 Bytes/s
HP Laserjet IIp	03:08 / 03:58 = 3.8 ppm	00:33 = 1.8 ppm	02:47 = 3.6 ppm	00:58	02:32	00:18 / 00:53	8.9 Pf.	*
NEC Silentwriter S60	00:33 / 02:49 = 5.3 ppm	00:24 = 2.5 ppm	01:54 = 5.3 ppm	00:49	01:20	00:22 / 00:46	7.9 Pf.	16800 Bytes/s
Okilaser 400	02:00 / 03:49 = 3.9 ppm	00:29 = 2.1 ppm	02:37 = 3.8 ppm	01:00	02:54	00:39 / 01:06	9.0 Pf.	8674 Bytes/s
Sanyo SPX-608	00:19 / 02:05 = 7.2 ppm	00:22 = 2.7 ppm	01:29 = 6.7 ppm	01:08	**	00:25 / 01:08	7.9 Pf.	6868 Bytes/s
Seikosha OP-105	00:23 / 02:58 = 5.1 ppm	00:13 = 4.6 ppm	01:57 = 5.1 ppm	01:23	07:27	00:48 / 01:01	7.4 Pf.	2370 Bytes/s

* Dieser Wert wurde beim HP nicht gemessen

** Der Speicherplatz des Sanyo reicht nicht aus

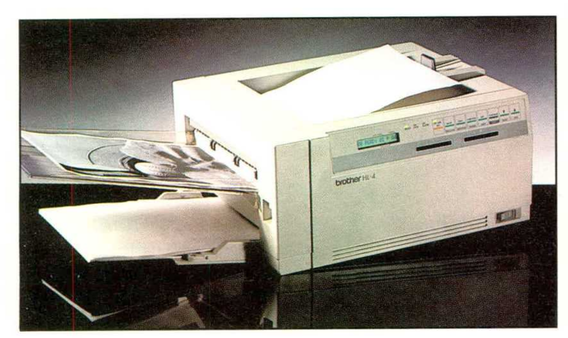
Die Moral von der Geschicht'...

...rat' niemand etwas Schlechtes nicht! Wenn Sie vor der Entscheidung stehen, sich einen Seitendrucker anzuschaffen, dann werden Sie die Tabelle, in der die Ergebnisse zusammengetragen sind, besonders kritisch betrachten. Sie sollten sich vor allem fragen, wozu Sie den Zukünftigen benutzen wollen, und mit welcher Software Sie drucken. Schreiben Sie viele Texte mit Programmen wie WordPlus oder benutzen Sie mit Script und Tempus Word die druckereigenen Schriften? Dann ist für Sie ein Gerät interessant, daß eine große Anzahl eigener Fonts besitzt wie z.B. der Brother HL-4. (Lesen Sie dazu auch den Extratext 'Freiheit der Schrift'.) Legen Sie dabei Wert auf hohe Geschwindigkeit? Dann sollten Sie den Seikosha OP-105 näher betrachten. Suchen Sie ein robustes Gerät für höchste Druckleistungen? Dann ist vielleicht der Sanyo SPX-608 für Sie richtig.

Zählen Sie jedoch zu den mehr Bildern und DTP zugewandten Zeitgenossen? Dann kann Ihnen der Schriftenreichtum schnurz sein, der befindet sich ohnehin auf Ihrer Festplatte. Sie brauchen gute Grafikqualität mit sauberen Schwärzen und scharfen Konturen. Da machen besonders der NEC Silentwriter2 S60 und der Oki-Laser 400 ein gutes Bild. Sind wiederum die großen Bilder nicht Ihr Ding, sondern eher die tägliche Arbeit mit schönen Pixel-Fonts? Dann haben Sie die freie Auswahl, denn dafür sind alle unsere Lichtdrucker gleich gut geeignet.

IB

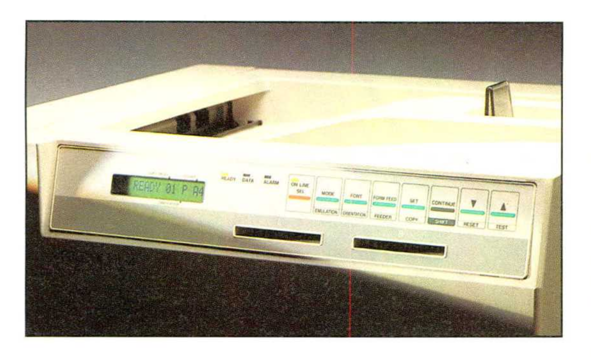
[1] Auf ein Neues - Test Atari SLM 605, ST-Computer 12/90 S.172ff.



Brother HL-4

Vieles von dem, was es zum Brother HL-4 zu bemerken gibt, gilt gleichermaßen für HPs LaserJet IIp. Sie basieren beide auf einem sehr kompakten Druckwerk (Engine) des japanischen Herstellers Canon. Entsprechend austauschbar sind die automatischen Papiereinzüge und die Belichtungs-Cartridges. Der Brother ist zum Quer-Aufstellen gedacht, den geringsten Raum verbraucht er dann, wenn der automatische Einzug, für den leider weitere 450 DM zu berappen sind, unter das Gerät geschraubt wird. Sein Bedienfeld zeigt sich nach Brother-Manier farbig und tastenreich. Die Folientasten haben einen guten Druckpunkt, und durchaus erfreulich ist, daß des Bruders Display auch die deutsche Sprache beherrscht. Leider fällt häufig ein Schatten darauf, so daß seine Lesbarkeit eingeschränkt ist. Aber dafür ist es in der Lage, einen Seitenzähler anzuzeigen, der über die Gesamtdruckleistung der Maschine informiert.

Sieht man vom Einzug ab, ist der kleine Japaner komplett ausgestattet. Neben der Kompatibilität zum LaserJet II emuliert er noch Epsons und IBMs Neunnadler sowie die Typenraddrucker von

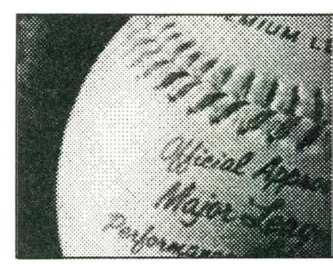


Diablo. Ebenso vollständig die Fonts: fünf verschiedene sind drin, davon zwei proportionale. Alle sind auch in Querrichtung zu verwenden - das ist durchaus nicht selbstverständlich.

Sein Handbuch ist gut gegliedert und illustriert, es fehlen jedoch Ausführungen zu den vielen Steuerbefehlen. Es ist eher anwenderorientiert gehalten. Die Wartung des Gerätes ist denkbar einfach - lediglich die Cartridge muß getauscht werden. Der Einzug zeigte sich unempfindlich gegenüber welligem Papier, das Wiederbenutzen ist also möglich. Daß der Anschlag auf dem Einzug das Druckgut nicht zentriert, scheint ihm nichts auszumachen, er zieht auch Briefumschläge ein. Leider wellt er sie so stark, daß sie praktisch unbrauchbar sind. Auch auf Folie will er nicht arbeiten: Der Druck verschmiert.

Seine Schriftqualität überzeugt. Die internen Fonts sind fein und gut lesbar, die Grafik hingegen gibt Anlaß zur Kritik. Zwischen Graustufen vermag er nicht korrekt zu trennen, manche Flächen sind gar schlierig. Das Schwarz ist nicht gut gesättigt: Auch komplett schwarze Flächen werden nicht als solche wiedergegeben.

Der besondere Gag am HL-4 ist 'DACT', die Technik, die Grafikdaten bereits beim Empfang komprimiert. Der Teilerfolg dieser Methode hinterläßt jedoch ein ungutes Gefühl. Nicht jede Ganzseitengrafik paßt in die virtuell vergrößerten 512 kB, und damit wird der Druck für den DTPler zum Vabanque-Spiel. Wer jedoch nur Strichzeichnungen mit viel Weiß zu Papier bringen will, hält es auch ohne nachgerüstetes Megabyte aus. Ein Lob verdient der Brother für sein sehr leises Auftreten, neben dem gut gearbeitet werden kann. 'He ain't heavy, he's my brother' sangen einst die Hollies, und das gilt auch für den kompakten Laserdrucker.

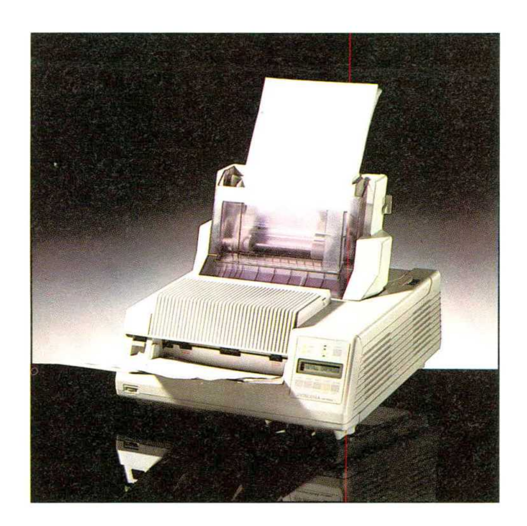


Testbild: Gerastertes Graustufenbild



Testbild: ST-Hardcopy - Strichzeichnung

Seikosha OP-105



Von japanischen Hersteller Seikosha ist man preiswerte und komplett ausgestattete Geräte gewohnt, und so verhält es sich auch mit dem OP-105, einem der günstigsten Geräte des Tests. Er ist ein LED-Drucker, wie der OkiLaser 400 auch. Daß die Drucktechnik keinen Einfluß auf die Qualität hat, zeigt sich beim Vergleich.

Leichter Frust kommt bei der Montage auf. Das Handbuch weist die nötigen Schritte zwar, doch hat es eher provisorischen Charakter. Die Abbildungen sind schlecht zu erkennen, das Werk ist recht unübersichtlich, und während der Arbeit blättert man immer wieder hilflos darin hin und her. Einen Index sucht man genauso vergeblich wie eine detaillierte Erklärung der Kommandos.

Freundlich ist die Beigabe der Plastikhandschuhe, damit beim Reinigen des Gerätes die Finger sauber bleiben. Doch sind sie unnötig: Die Wartung des Seikosha ist einfach und nicht schmutzig: Lediglich die Cartridge ist alle 6.000 Blatt zu erneuern. Der OP-105 ist ein wenig ungewöhnlich aufgebaut: Er druckt von hinten nach vorn. Das Papier wird dabei kaum gebogen, klaglos nimmt er auch besondere Materialien an. Nur mit Briefumschlägen mag er sich nicht anfreunden - die zerknitterten Resultate lassen zum Aufkleber greifen. Papier besorgt er sich aus einem Vorrat, der seinen Platz, ähnlich wie bei Nadeldruckern, stehend auf der Maschine hat. Für 440 DM ist noch ein zweiter Schacht für eine weitere Papiersorte etc. erhältlich. Leider erwies sich der Einzug als recht pingelig. Das Gerät zieht zwar zügig ein, doch häufig bemängelt es dabei auch neues Papier.

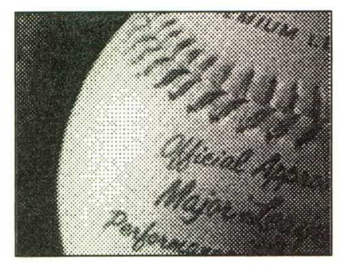
Der manuelle Einzug befindet sich ebenfalls oben, direkt vor dem Einzug. Seine Handhabung wird dadurch unnötig erschwert, daß man zunächst das Druckermenü aufrufen und die Handzuführung wählen muß. Eine Face-Up-Ablage erscheint bei dieser Form des Druckerbaus unnötig, folgerichtig hat der Seikosha auch keine.

Die Kommunikation mit der Maschine geht einfach vonstatten, das übersichtliche Display stellt einen häufig vor die Wahl 'Clear Buffer?', die es dann über die leider klein geratenen Folientasten zu treffen gilt. Insgesamt ist die Bedienung problemlos - sogar der Tonerregler findet sich an der Front des Geräts, und der Netzschalter sitzt löblicherweise obendrauf. Einzig die ewige Pieperei bei Fehlern oder als Quittung für einen Tastendruck nervt.

An Ausstattung läßt der Seikosha OP-105 keine Wünsche offen. Er bietet die größte Font-Auswahl unter den hier getesteten Geräten. Dabei sind alle auch quer zu drucken - der Gestaltung sind kaum Grenzen gesetzt. Reichhaltig ist ebenfalls die Emulationsauswahl. Wie der Brother versteht

der OP-105 neben denen des HP LaserJet II auch die Kommandos der Epson- und IBM-Neunnadler sowie die der Diablo-Drucker.

Serienmäßig wird der Seikosha mit 512 kB ausgestattet, für Grafikanwendungen ist jedoch die Erweiterung auf 1,5 MB unabdingbar. Trotzdem druckte er den grafischen Brief klaglos auch mit dem geringen Speicher, was z.B. der Oki sofort ablehnte. Bei der Geschwindigkeit des Textdrucks mischt der OP-105 in der vorderen Hälfte des Feldes mit, bei der Grafik verlassen ihn die Kräfte. Ganz offensichtlich ist sein Betriebssystem mit größeren Datenmengen überfordert. Die Geschwindigkeit, mit der er die Daten abnimmt, sinkt dann stark. Beim Thema Textausgabe ist er der einzige Prüfling, der die vom Hersteller angegebene Geschwindigkeit noch übertraf. Die Qualität seiner internen Schriften ist gut, bei manchen fällt der etwas fette Strich auf. Die Grafikqualität rangiert ebenfalls im Mittelfeld. Der kräftige Strich und die schlecht nuancierten Grauwerte trüben das Bild.



Testbild: Gerastertes Graustufenbild



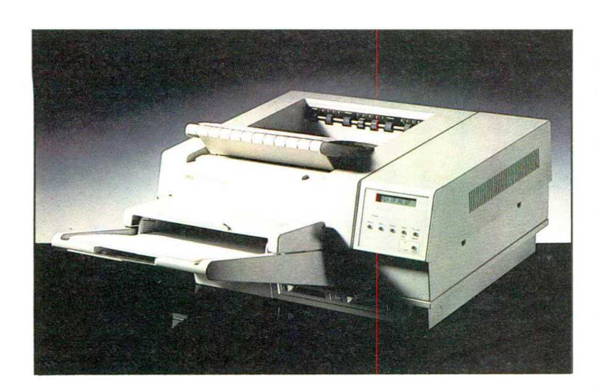
Testbild: ST-Hardcopy - Strichzeichnung

Der Seikosha OP-105 zeigt der Konkurrenz, daß ein komplett ausgestatteter Drucker nicht teuer sein muß. Er hat mit 7,4 Pfennigen pro Seite die niedrigsten Verbrauchskosten des Tests. Leider muß man bei der Grafikgeschwindigkeit und im Papier-Handling Abstriche machen, doch insgesamt ist er das Gerät mit dem besten Preis/Leistungsverhältnis.

NEC Silentwriter2 S60

Zuerst sahen wir ihn auf der Orga'90 - den neuen NEC. Er basiert auf einer 6-Seiten-Engine von Minolta und zählt zu den massiven Geräten dieses Vergleichs. Seine fast 20 kg Lebendgewicht deuten darauf hin, daß Metall in ihm verbaut wurde, ja sogar Teile des Gehäuses bestehen daraus. Der NEC bietet einen fast geraden Durchlauf des Papiers, für besondere Zwecke kann an seiner Rückseite eine Face-Up-Ablage angebracht werden. Wer diese Option nicht häufig nutzt, kann sich die 270 DM, die sie kostet, auch sparen, die Klappe abschrauben und das Blatt einfach rausfallen lassen. An der Vorderseite befindet sich die Papierkassette, die über zentrierende Führungen verfügt. So werden auch kleinere Formate immer mittig eingezogen. In die Kassette wurde der manuelle Einzug eingebaut. Praktisch ist, daß bei entsprechender Menüeinstellung der Drucker selbst erkennt, wenn ein Papier dort anliegt, und es dann einzieht.

Die Einstellung und Bedienung des Gerätes ist sehr einfach - die Kommunikation über das auch der deutschen Zunge mächtige Display geht reibungslos vonstatten. Die Wartung wird im Handbuch gut beschrieben. Neben dem Tausch der Belichtungseinheit muß auch ein Filz in der Fixiereinheit ersetzt und einer der Ladedrähte gereinigt werden. Darüber hinaus weist

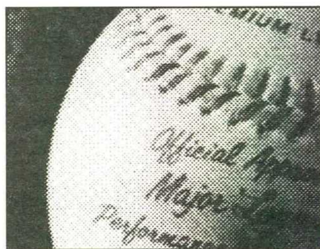


das umfangreiche und gut illustrierte Handbuch viel Information auf. Die Steuerbefehle sämtlicher Drucker sind ausführlich und mit Beispielprogrammen aufgeführt. Das alles findet seinen Platz in einem DIN A5-Ringbuch.

Positiv ist der relativ geringe Seitenpreis von 7,9 Pfennigen, der aus der hohen Standzeit der Cartridge resultiert. Bei der Ausstattung an Fonts ist der NEC S60 das Schlußlicht des Vergleichs. Den Standard-Courier-Font kennt er nicht einmal in kursiv. Außer dem LaserJet II emuliert er einen Diablo-Typenraddrucker und bietet den 'NEC Native Mode', einen eigenen Kommandosatz. In der HP-Emulation hatte auch er eine andere Vorstellung vom oberen Rand, den er gleich zwei Zeilen höher setzt als der LaserJet II. Obwohl er für alle Druckmedien gut geeignet ist, scheiterte der Versuch, mit ihm auf Karton zu drukken, bereits am Einzug. Erfreuliches gibt es von der Papierablage an der Oberseitezu berichten: Sie hält Platz für viel Papier bereit.

Paradoxerweise bietet uns NEC mit dem 'Silentwriter' das lauteste Gerät des Tests. Der Lüfter ist an der Grenze des Erträglichen und bedarf der Überarbeitung. Ungeschickt ist die Anbringung des winzigen Netzschalters an der Rückseite der Maschine.

Qualitativ ist der NEC zur Spitzengruppe zu zählen. Seine Grafik ist die beste des Vergleichs, ganz knapp gefolgt vom Oki. Sie gibt keinerlei Anlaß zur Kritik. Lediglich bei CAD-Anwendungen könnte der sehr feine Strich eher von Nachteil sein. Auch seine internen Fonts sind sehr fein gezeichnet, vielleicht schon zu dünn. Was den NEC Silentwriter2 S60 zu einer wirklich faszinierenden Maschine macht, ist seine Geschwindigkeit. Weniger im Textbetrieb, hier erreicht er sein Soll von sechs Seiten in der Minute nicht ganz. Bei der Abnahme grafischer Daten arbeitet er jedoch so schnell wie kein anderes Gerät. Innerhalb von 1,5 Minuten bringt er fast ein ganzes Megabyte an Daten zu Papier. Voraussetzung ist allerdings, daß die Software entsprechend schnell ist. So setzt er sich im grafischen Teil der Zeitmessungen an die Spitze des Testfeldes. Der NEC verfügt von Haus aus über eine Speicherausstattung von einem Megabyte. Da paßt eine Ganzseitengrafik auch hinein, so daß die kostspielige RAM-Erweiterung nicht nötig wird. In letzter Minute erreicht uns die Nachricht, daß im Rahmen der Modellpflege die nächsten S60 über einen geregelten Lüfter verfügen werden - dieser Kritikpunkt ist also bereits erhört worden. Der S60 besticht durch seine hohe Geschwindigkeit und die saubere Qualität.



Testbild: Gerastertes Graustufenbild



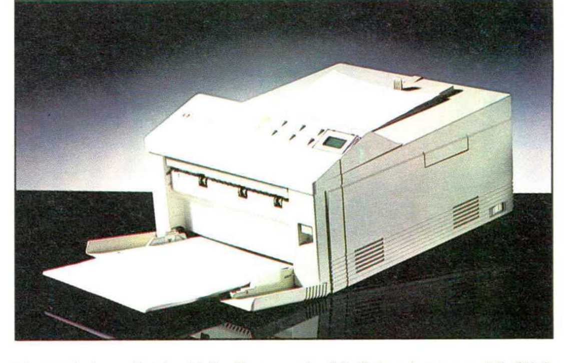
Testbild: ST-Hardcopy - Strichzeichnung

Hewlett Packard LaserJet IIp

Wie der Brother HL-4, basiert der HP LaserJet IIp auf einer 4-Seiten-Engine von Canon. Dadurch sind seine mechanischen Eigenschaften identisch. Auch hier wird das Druckgut beim Weg durch die Maschine stark hin- und hergebogen, so daß besonders Umschläge zu stark geknittert werden. Aber auch alle anderen Papiere verlassen den Drucker gewellt und verlangen nach waagerechter Lagerung.

Die Wartung des HP reduziert sich ebenfalls auf den Tausch der Belichtungseinheit, was denkbar einfach vonstatten geht. Das Handbuch ist geradezu vorbildlich, was die Montage des Gerätes betrifft. Um kleine Störungen zu beheben, für die Sorgen des täglichen Anwenders, hat es die Antworten schnell parat. Das ist Tradition bei HP. Allerdings auch, daß dem Käufer jegliche technische Dokumentation praktisch vorenthalten bleibt. Wer seinen IIp programmieren will, der kaufe sich für 50 DM den 'Technical Reference Guide', der in die hehren Weihen der PCL-Programmierung einweist.

Leider ist noch ein anderes Problem von HP-Druckern bekannt. Das taucht aber nur beim Anschluß am Atari auf. Denn ein HP-Gerät muß - sonst druckt der Atari ins Leere - partout vor dem Rechner eingeschaltet werden. Gegen diese Eigenheit haben findige Anbieter mittlerweile eine kleine Box zum Aufstecken parat. Dummerweise aber wertet der Atari nicht die laut Centronics-Standard als 'OnLine' bekannte Leitung aus. Er mißbraucht dazu die Busy-Leitung, und das hat fatale Folgen. Bekommt der HP LaserJet IIp nämlich mehr als drei Seiten Text auf einen Schlag, so bricht er während der Übertragung der vierten Seite ab und meldet sich für mehr als 30 Sekunden ausschließlich 'busy'. Währenddessen druckt er die ersten drei Seiten, um dann für wenige Sekunden wieder empfangsbereit zu sein. Bis zum Ende der Übertragung holt er sich die Daten immer nur in Häppchen von wenigen Sekunden Dauer.



Die BIOS#3-Funktion, die ein Zeichen auf die Schnittstelle ausgibt, wartet aber nur 30 Sekunden auf den Drucker und bricht dann



Kompaktes Innenleben: Blick in den HPIIp

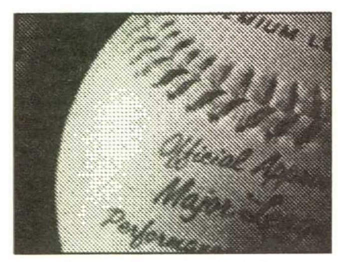
ohne Fehlermeldung ab. Es geht aber nicht nur das eine Zeichen verloren. Bis zur nächsten Änderung des Busy-Pegels bricht sie jeden weiteren Versuch einer Übertragung sofort ab - die Zeichen werden ins Nichts gesandt. Einem Programm, das den Druckerstatus nur vor der Übertragung einmal abfragt und dann ganze Blöcke druckt, gehen so Teile verloren. Dagegen hilft nur, vor jedem zu sendenden Zeichen die Bereitschaft des Druckers zu prüfen und ggf. darauf zu warten. Das alles liegt daran, daß sich Atari die eine Leitung für das 'OnLine'-Signal gespart hat. Allerdings fand ich kein Textprogramm, das auf dieses Verhalten des Druckers hereingefallen wäre.

Die Bedienung des HP IIp ist sehr einfach. Die großen Tasten auf der Oberseite ermöglichen die nötigen Funktionen, Voreinstellungen werden über das - recht kleine - Display, das auch deutsch spricht, getätigt. Daß er nur zum LaserJet II kompatibel ist, ist verständlich, aber bei den Schriften ist die Auswahl ebenso dürftig. Lediglich Courier in zwei Größen sowie den LinePrinter hält er bereit, ist allerdings auch in der Lage, diese quer zu drucken.

Bleibt die Druckqualität zu erwähnen. Hier zeigen sich trotz des identischen Druckwerks Unterschiede zum Brother. Die Fonts des HP sind recht fett, sie können nur befriedigen. Die Grafikqualität zeigt mangelhafte Schwärzen, graue Flächen werden häufig schlierig und zu den Kanten hin werden gefüllte Flächen dunkler. Ebenfalls fällt ihm die Grauabstufung schwer.

Die PCL-Testseite zeigte, daß ausgerechnet der HP einen Befehl anders interpretiert, als es der LaserJet II tut. Da er bereits in der Zeile eins drucken kann, beginnt er auch dort mit der Zählung der Zeilen. Ein gesetzter oberer Rand landet bei ihm eine Zeile höher als beim Vorbild.

Der IIp ist der kleine Bruder der Standardmaschine HP Laserjet II, und für ihn spricht mit Sicherheit das reichhaltige Zubehör, das es auch von Fremdanbietern gibt. Außerdem ist er ein echter Leisetreter und leicht zu bedienen.



Testbild: Gerastertes Graustufenbild



Testbild: ST-Hardcopy - Strichzeichnung

OkiLaser 400

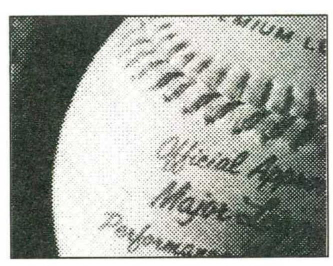


Mit dem OkiLaser 400 stellt uns der japanische Hersteller den kleinsten Vertreter seiner Reihe von LED- (nicht Laser!) Druckern auf den Tisch. Das Gerät ist fast klassisch gebaut - es hat dabei Ähnlichkeiten mit dem NEC. Das Papier wird aus einer geschlossenen Kassette, ähnlich wie bei Photokopierern, eingezogen und auf seinem Lauf durch die Maschine lediglich einmal um 180 Grad gekrümmt. Das läßt sich dadurch ändern, daß man die Klappe an seiner Rückseite öffnet und das Papier Face-Up ausgibt. Mit dieser Öffnung erschließt sich der OkiLaser das gesamte Spektrum an Druckmedien. Aufkleber, Umschläge, alles kein Problem. Der Druck auf Folie läßt sich sehen.

Einzig der Kasetteneinzug ist recht pingelig. Papier, das schon einmal durch den Drucker gelaufen ist, weist er häufig mit rüden Papierstaus ab. Positiv fällt die manuelle Einzelblattzuführung auf. Sie besteht aus zwei zentrierenden Führungen auf der Papierkassette. Wenn ein Blatt eingelegt wurde, erkennt der Drucker es automatisch und zieht es als nächstes ein.

Das Handbuch zum Oki besteht aus zwei Teilen. Zum einen dem Druckerhandbuch, das die Aufstellung, die Montage und Inbetriebnahme sowie die Wartung gut illustriert beschreibt. Beim Oki haben wir es mit einer Maschine zu tun, deren Bildtrommel nur in Abständen von 12.000 Blatt getauscht werden muß. Der Toner ist häufiger fällig und wird einzeln nachgefüllt. Leider spiegelt sich das nicht im Seitenpreis wider, da ist er der Spitzenreiter. Das zweite, das Benutzerhandbuch, hingegen behandelt mehr die softwaretechnische Seite. Hier werden viele PCL-Befehle nur erwähnt, Stichwortverzeichnisse fehlen beiden Werken, und eine gewisse Unübersichtlichkeit wohnt ebenfalls beiden inne. Trotzdem kommt man schnell mit dem OkiLaser klar, das große und gut ablesbare Display hilft dabei. Zusammen mit den Folientasten, die einen ordentlichen Druckpunkt bieten, läßt sich der Drucker gut handhaben.

Neben der Kompatibilität zum HP LaserJet II bietet der OkiLaser auch noch die Emulation eines Epson FX und eines Diablo-Typenraddruckers an. Seine Font-Ausstattung kann sich sehen lassen. Besonders schön: die in zwei Höhen verfügbare Times-Schrift, die zudem proportional ist. Leider ist das Angebot an quer zu drukkenden magerer.



Testbild: Gerastertes Graustufenbild

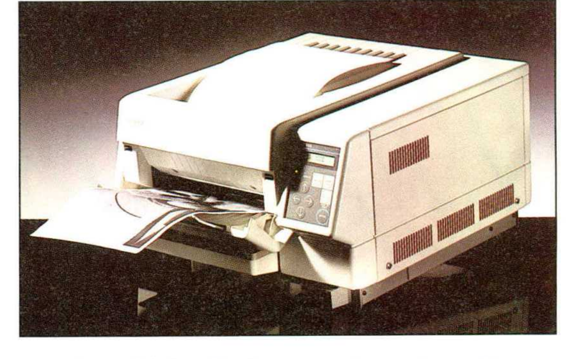
Von der Qualität ist wiederum nur Gutes zu berichten: Die Schriften sind scharf und mit feinem Strich gezeichnet. In der Grafikqualität liegt der Oki nur einen Hauch hinter dem Nec - auch an seinen Druckerzeugnissen gibt es nichts auszusetzen. Die Geschwindigkeit, die er im Textdruck zeigt, ist ebenso kein Anlaß zur Kritik.



Testbild: ST-Hardcopy - Strichzeichnung

Der Oki ist zwar eines der kleineren Geräte im Test, setzt aber vor allem beim Papier-Handling und bei der Druckqualität Maßstäbe. Wermutstropfen bleibt seine überdurchschnittlich hohe Druckzeit im Grafikmodus. Lob verdient der 'Sleep-Modus', der ihn nach einer einstellbaren Zeitspanne deutlich weniger Geräusch produzieren läßt.

Sanyo SPX-608

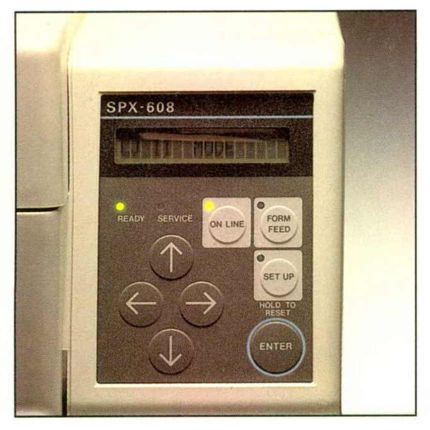


In vielerlei Hinsicht ist der Sanyo SPX-608 eine beeindruckende Maschine. Beginnend bei den Ausmaßen des Gehäuses, über die 22 kg, die er auf die Waage bringt, bis zu seinem Aufbau. Wie bereits gesagt, ähnelt er stark einem Kopierer. Die Komponenten seines Druckwerkes sind von links her zugänglich und können sämtlich durch Herausziehen getauscht und gereinigt werden.

Das Handbuch besticht durch seine Aufmachung (ein DIN A5-Ringbuch) und Übersichtlichkeit. Behutsam und gut illustriert werden alle Schritte zum Aufbau des Geräts erklärt. Ebenso findet die Wartung einen angemessenen Platz. Die ist beim SPX-608 nicht ganz so einfach wie bei den anderen Maschinen. Der Tonerbehälter befindet sich in einer eigenen Einheit, das alles muß beim Tausch gereinigt werden. Ebenso wie die beiden Ladungsdrähte, die herausgezogen werden müssen. Da sich im Laufe der Zeit Tonerreste auch im Innern der Maschine absetzen, bleiben die Hände dabei nicht immer sauber. Leider wandert das reichliche Material komplett in den Müll. Vorbildlich jedoch ist wiederum die Bedienungsanleitung im Innern der Klappe.

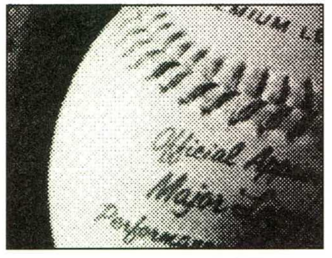
Daß der Sanyo SPX-608 für Leute gebaut ist, die es ernst meinen, zeigt sich, wenn man die Verpackungseinheiten für das Verbrauchsmaterial sieht: In einem großen Karton befinden sich gleich Toner und die entsprechende Einheit für 15.000 Kopien. Durch den günstigen Preis erreicht der Sanyo einen relativ niedrigen Seitenpreis. Die Bildtrommel muß gesondert geordert werden.

Leichten Zugang zum Drucker findet der Anwender über das Display, das in Verbindung mit den gut zu bedienenden Tasten die Bearbeitung des internen Menüs einfach macht. Lästig ist, den SPX-608 vor jeder manuellen Papier-



Das Bedienfeld des Sanyo

lieferung davon in Kenntnis setzen zu müssen. Das normale Papier wird aus einer Kassette, die sich unten im Drucker befindet, eingezogen. Obwohl das Druckgut so häufig gebogen wird, funktionierte der Einzug einwandfrei, wenngleich er bei welligem Papier leicht ins Hakeln gerät. Nebeneffekt dieser Anordnung ist, daß das Gerät sehr kompakt bleibt.



Testbild: Gerastertes Graustufenbild

Zusätzlich zum manuellen Einzug an der Rückseite kann auch an der Front eine Klappe geöffnet werden. So wird das Druckgut überhaupt nicht mehr gebogen, und der SPX-608 druckte klaglos auf alle Materialien, die wir ihm boten.



Testbild: ST-Hardcopy - Strichzeichnung

Der Sanyo - das zeigen die Druckzeiten -findet sein Betätigungsfeld hauptsächlich im Textdruck. Da ist er bei weitem Spitzenreiter des Vergleichs - wenngleich bei der Schriftauswahl herbe Abstriche zu machen sind. Lediglich die beiden nötigsten Fonts hat er zu bieten. Die Schriftqualität ist recht fett, man sollte sich mit feineren Download-Fonts behelfen. Die Grafikqualität verdient mehr Kritik: schlieriges Grau, mangelnde Schwärzung und unschöne Zeiligkeit ziehen sich durch alle Bilder. Daran zeigt sich, daß der SPX-608 als relativ altes Modell für dergleichen Anforderungen nicht konzipiert ist.

Auch der Sanyo bietet eine Datenkompression ankommender Grafikdaten. Doch obwohl laut Hersteller dabei der 1 MB große Standardspeicher auf virtuelle 1,5 MB vergrößert werden soll, paßte noch nicht einmal das Testbild mit seinen 970 kB hinein. Schade, denn wenn man für Ganzseitengrafik nicht die teuere RAM-Erweiterung kaufen müßte, wäre er das billigste Komplettgerät im Test.

Neben seiner beeindruckenden Textgeschwindigkeit erfreut der Sanyo SPX-608 mit Details wie dem von außen ablesbaren Seitenzähler und seinem Handbuch, das fast ein PCL-Lehrwerk ist. Leider müssen in puncto Qualität - vor allem bei der Grafik - Abstriche gemacht werden.

IB

Hersteller	Brother	Hewlett-Packard	NEC	Oki	Sanyo	Seikosha
Typ	HL-4	LaserJet IIp	Silentwriter2 S60	Okilaser 400	SPX-608	OP-105
PREISE inkl. MWSt						
Grundgerät	3933 DM	3933 DM	4394 DM	2998 DM	3409 DM	2998 DM
Speichererweiterung	1 MB: 1083 DM	1MB: 804 DM	1MB: 1254 DM	1 MB: 698 DM	2 MB: 1382 DM	1 MB: 949 DM
Font-Karten	je 455-569 DM	je 200-400 DM	250 DM	348 DM	411 DM	215 DM
Emulationskarte	HPGL: 683 DM	PS: 1408 DM	HPGL: 1850 DM	-	-	-
Papiereinzug/-kassette	455 DM	451 DM	inklusive	inklusive	inklusive	inklusive
serielle Schnittstelle:	inklusive	inklusive	inklusive	Gerät auch mit ser. lieferbar	inklusive	inklusive
Trommel	211 DM	240 DM	341 DM	598 DM	410 DM	358 DM
Standzeit Trommel	3500 Blatt	3.500 Blatt	6.000 Blatt	12.000 Blatt	15.000 Blatt	6.000 Blatt
Toner	(eine Einheit	(eine Einheit	(eine Einheit	68 DM	564 DM *	(eine Einheit
Standzeit Toner	mit Trommel)	mit Trommel)	mit Trommel)	2.500 Blatt	15.000 Blatt	mit Trommel)
Standzeit Gerät	180.000 Blatt	180.000 Blatt	180.000 Blatt	180.000 Blatt	180.000 Blatt	180.000 Blatt
Seitenpreis	8,1 Pfennige	8,9 Pfennige	7,9 Pfennige	9,0 Pfennige	7,9 Pfennige	7,4 Pfennige
Garantie	12 Monate	12 Monate	12 Monate	12 Monate	6 Monate	6 Monate
AUSSTATTUNG						
Drucktechnik, Auflösung	Laser, 300 DPI	Laser, 300 DPI	Laser, 300 DPI	LED, 300 DPI	Laser, 300 DPI	LED, 300 DPI
Emulationen	HP Laserjet II	HP Laserjet II	HP Laserjet II	HP Laserjet II	HP Laserjet II	HP Laserjet II
	Diablo		Diablo	Diablo	Diablo	Diablo
	Epson FX		Nec Native Mode	Epson FX	Epson FX	Epson FX
	IBM Proprinter XL					IBM Proprinter XL
Fonts hochkant/Punkthöhe	Courier 10	Courier 10	Courier 10 2	Courier 7		
(jew. auch fett und kurs.)	Courier 12	Courier 12	Courier 12 1	Courier 12	Courier 12	Courier 12
	LinePrinter 8,5 3	LinePrinter 8,5 3	LinePrinter 8 3	LinePrinter 8 3	LinePrinter 8 3	LinePrinter 8 3
	Times 10 prop			Times 10 prop		Times 10 prop
				Times 8 prop		
	Helvetica 10 prop			Helvetica 14 prop 3	Helvetica 10 prop	
					Prestige El. 10 2	Prestige El. 10
						Prestige El. 6
						Gothic 7,9.5,12,14

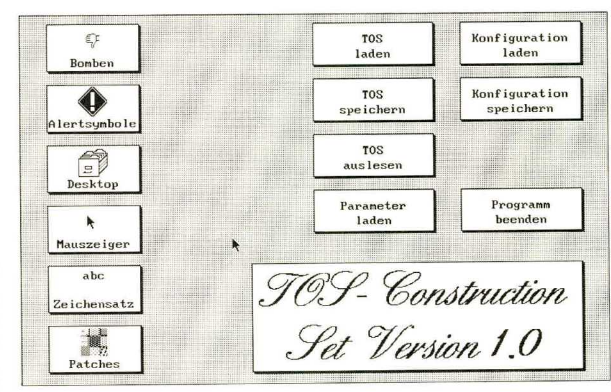
Fonts quer/Punkthöhe (jew. auch fett und kurs.)	alle	alle	Courier 10 1 LinePrinter 8 3	Courier 12 1 LinePrinter 8 3	Courier 12 1 LinePrinter 8 3	alle
Speichergröße	512 kB + Kompr.	512 kB	1024 kB	512 kB	1024 kB + Kompr.	512 kB
Schnittstellen	parallel/seriell	parallel/seriell	parallel/seriell	je nach Ausführung	parallel/seriell	parallel/seriell
TESTS						
1. mechanische Beurteilung:						
automatischer Einzug	gut	gut	gut	neigt zum Hakeln	gut	neigt zum Hakeln
manueller Einzug	gut	gut	gut	sehr gut	schlecht (hinten)	befriedigend
Face-Up-Ablage	gut	nicht vorhanden	gegen Aufpreis	sehr gut	befriedigend (kurz)	nicht vorhanden
Druck auf Umschläge	knittern stark	knittern stark	gut	sehr gut	sehr gut	schlecht (hakeln)
Druck auf Folie	ungeeignet	ungeeignet	geeignet	gut	gut	ungeeignet
Aufkleberdruck	sehr gut	sehr gut	sehr gut	sehr gut	sehr gut	sehr gut
2. qualitative Beurteilung:						
interne Schriften	sehr gut, feiner Strich	befriedigend, fetter Strich	sehr gut, sehr feiner Strich	sehr gut, feiner Strich	befriedigend, fetter Strich	befriedigend, fetter Strich
grafischer Druck	gut, Graumuster sind schlierig, Grauwerte mäßig nuanciert, Schwarz schlecht gesättigt	befriedigend, Graumuster sind schlierig, Grauwerte schlecht nuanciert, Schwarz schlecht gesättigt	sehr gut, feiner Strich	sehr gut, feiner Strich, andere Füllmuster als Original HP	unbefriedigend, unscharf, Flächen zeilig, Schwarz schlecht gesättigt	gut, Grauwerte schlecht nuanciert, fetter Strich
Text mit Pixel-Fonts (grafisch)	sehr gut, für einige Fonts zu dünn	sehr gut	sehr gut, für einige Fonts zu dünn	sehr gut, für einige Fonts zu dünn	sehr gut	sehr gut
Bemerkungen	kleines, kompaktes Gerät mit guter Ausstattung und Qualität. Papiermagazin nur gegen	kleines, kompaktes Gerät. Einfach zu bedienen. Papiermagazin nur gegen Aufpreis	massives, schnelles Gerät mit sehr guter Qualität. 1 MB serienm., sonst magere Ausstattung	kleines, preiswertes Gerät mit gutem Papier-Handling. Sehr gute Qualität.	massives Gerät mit hoher Geschwindigkeit im Textdruck.	preiswertes Gerät mit vielen Fonts. Geringe Verbrauchskosten. Gutes hältnis.

1 Dieser Font nicht kursiv
2 Dieser Font nicht fett
3 Dieser Font nicht fett und nicht kursiv

* Für den Sanyo ist ein Copy-Kit mit 5 Tonerpatronen für je 3000 Blatt erhältlich.

Toss your TOS

oder: blühe, Desktop!



Die Zeiten des eintönig grauen Desktops, der Arbeitsumgebung eines jeden ST-Besitzers, sind seit einigen Monaten vorbei. Der individuelle ST hat Einzug in die Computerzimmer gehalten. Abwechslung tut gut! Das dachte man sich auch bei Rosin Datentechnik und brachte ein „TOS-Construction-Set" auf den Markt. Was sich hinter dieser abenteuerlichen Bezeichnung verbirgt, soll der folgende Artikel zeigen.

Wenn Sie keinen TT oder STE haben, werden Sie das Gefühl sicher kennen, wenn Sie schon wieder auf die eintönigen Laufwerkssymbole blicken und der ewig gleiche Zeichensatz Sie nervt. Und wenn Sie den Rechner anschalten, dauert es eine Ewigkeit, bis Sie tatsächlich mit ihm arbeiten können. Dank der Fehler im Betriebssystem starten Sie nämlich Dutzende von Autostartprogrammen, die die Fehler ausmerzen sollen. Es ist einleuchtend, daß dieser Zustand so nicht anhalten kann und sich jemand um die Behebung kümmern muß.

Nun ist es - wie gesagt - möglich, dem Rechner mit Dutzenden von Autostartprogrammen die Fehler auszutreiben. Mit anderen Programmen ist es wiederum möglich, dem Desktop ein wenig auf die Sprünge zu helfen - etwa durch neue Icons (also Laufwerkssymbole oder Mülleimer) oder einen neuen Zeichensatz. Diese Möglichkeiten haben eines gemeinsam: Sie sind nur mit residenten Programmen durchführbar (Programme, die einmal gestartet werden und dann im Hintergrund verweilen) und kosten zusätzlichen Speicherplatz, von dem man bekanntlich nie genug haben kann.

Die Lösung

Diesem Problem beizukommen, ist eigentlich ganz einfach. Doch wie das nun mal mit einfachen Lösungen ist, kommt man nur sehr schwer auf sie. Trotzdem bleibt es einfach: Alles, was man als Laie nur schwer ändern kann, könnte doch eigentlich durch ein Programm erledigt werden! Ein solches Programm liegt jetzt vor: das *TOS-Construction-Set* in der Version 1.0. Was mit dem Programm möglich ist, ist schnell erklärt. Zum einen lassen sich alle Desktop-Icons auswechseln. Damit hat die tägliche Tristheit ein Ende. Zum anderen lassen sich Patches am Bestriebssystem anbringen, also Verbesserungen von Fehlern oder auch Erweiterungen vorhandener Routinen.

Dieser Vorgang ist nun recht einfach, denn das TOS-Construction-Set ist durch und durch menügesteuert. Alles, was man an Aktionen durchführen kann, wird mit der Maus erledigt. Das geht in diesem Fall am schnellsten und macht am wenigsten Arbeit. In Bild 1 sehen Sie die Möglichkeiten, die das TOS-Construction-Set bietet. Zum einen lassen sich die Symbole für Bomben verändern. Wesentlich wichtiger sind jedoch die folgenden Funktionen. So lassen sich beispielsweise die Symbole der Alert-Boxen ebenso wie alle Desktop-Icons frei edieren. Wem dann auch der Mauszeiger noch zu langweilig ist, der kann seiner Phantasie freien Lauf lassen und auch diesen verändern. Zu guter Letzt läßt sich auch der langweilige Atari-Zeichensatz gegen einen neuen auswechseln. In den Bildern 2 bis 4 sehen Sie die Änderungsmöglichkeiten im Überblick.

So geht's

Um nun sein eigenes Desktop zu erstellen, benötigt man das TOS als Datei - entweder als 1x192- (ladbar), 3x64- (EPROMs) oder 6x32-kB-Dateien (ebenfalls EPROMs). Weiterhin benötigt man eine Parameterdatei, in der angegeben ist, an welchen Speicherstellen welche Dinge (wie z.B. Icons) zu finden sind. Für TOS 1.04 wird diese Datei bereits mitgeliefert. Das einzige, was man nun noch benötigt, ist das TOS auf Diskette, das man sich am besten aus seinem eigenen Rechner auslesen und auf ein Medium abspeichern kann. Weiter geht's, indem man die Funktion *TOS auslesen* betätigt. Dadurch sucht sich das Programm automatisch alle Icons heraus, die in der TOS-Datei enthalten sind. Ist das geschehen, ist es an der Zeit, die passenden Icons zu laden bzw. zu edieren. Zum Laden stehen genügend Bilder zur Verfügung, denn sie werden mitgeliefert. Natürlich darf man aber auch jegliche anderen Bilder benutzen, sofern sie im Standardformat vorliegen.

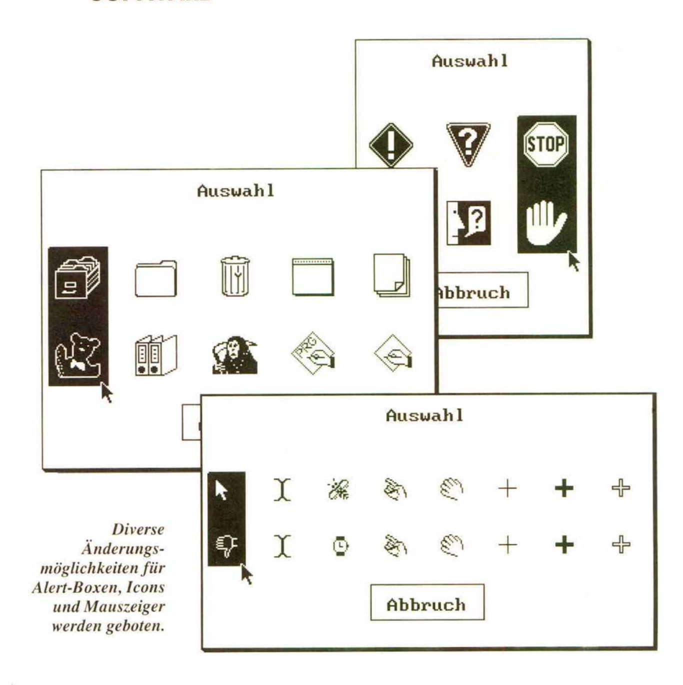
Zum Edieren der Icons steht ein ausreichend komfortabler Editor zur Verfügung, der leider aber nicht mit Turbo ST funktioniert. Hier zeigt sich auch eines der wenigen Mankos des Programms, denn es besitzt keine Menüleiste. So muß man vorher daran denken, alles abzuschalten und/oder einzustellen, was man benötigt. Eigentlich schade, wenn man bedenkt, wie einfach eine Menüleiste doch zu realisieren ist.

Konfigurationen

Die Konfiguration kann natürlich auch abgespeichert und wieder eingeladen werden, um spätere Änderungen komfortabel durchführen zu können.

Nun beschränkt sich das TOS-Construction-Set natürlich nicht nur darauf, neue Icons einzubauen, sondern es ist auch möglich, wichtige Änderungen am Programmteil des Betriebssystems vorzunehmen, allgemein unter dem Namen „Patch“ bekannt. Mitgeliefert werden einige Patches für die TOS-Version 1.04 (vormals 1.4), etwa eine Verbesserung der Uhrabfrage nach einem Reset, Fastload oder die serielle Schnittstelle, um nur einige zu nennen. Eigene Patches sind auch recht einfach einzubauen, denn sie müssen lediglich in einer ASCII-Datei abgelegt werden, deren Aufbau zwar nicht in der Anleitung beschrieben, jedoch recht leicht zu durchschauen ist. Die verschiedenen Änderungen werden in einem Pop-Up-Menü aufgezeigt und lassen sich hier selektieren und auch wieder deselektieren.

Wenn alle Änderungen zur vollen Zufriedenheit vorgenommen sind, sollte man sein TOS abspeichern. Hier bieten sich wieder verschiedene Möglichkeiten an. So ist es möglich, das TOS als eine Datei abzuspeichern, die nachher durch ein Ladeprogramm nachgeladen werden kann. Wenn man sich sein neues Betriebssystem lieber gleich auf EPROMs brennen möchte, weil man nicht so viel Speicherplatz oder keine Lust hat, dauernd auf das Laden zu warten, existiert natürlich auch hier die Möglichkeit, das TOS als aufgesplittete Dateien abzuspeichern.



Diverse Änderungsmöglichkeiten für Alert-Boxen, Icons und Mauszeiger werden geboten.

Autoload

Wer nicht die Möglichkeit hat, sich sein „neu“ entstandenes TOS auf EPROMs zu brennen und gegen das originale Betriebssystem auszutauschen, kann das TOS auch nachladen - entweder von Diskette oder von Festplatte. Dazu muß die TOS-Datei aber erst nachbearbeitet werden, sie wird „reloziert“. Für diesen Zweck liefert Rosin ein PD-Programm mit, mit dem diese Funktion problemlos ausgeführt werden kann. Ist diese Arbeit erledigt, legt man lediglich ein (ebenfalls mitgeliefertes) Startprogramm in den Auto-Ordner und die nunmehr relozierte TOS-Version in das Wurzelverzeichnis, von wo aus sie beim Neustart geladen wird und fortan auch resetfest ist.

Das TOS-Construction-Set ist ein interessantes Programm, mit dem man seinem tristen Desktop noch einmal auf die Sprünge helfen kann. Lediglich das Handbuch verdient einige Schelte. Viele Tatsachen, die man beispielsweise zum Nachladen des TOS benötigt, werden einfach verschwiegen. So ist es unverständlich, daß die Relozierung des TOS nicht mit einem Wort angesprochen wird. Ein weiteres Manko ist die fehlende Menüleiste, die man aber verschmerzen kann. Ebenfalls unschön ist die fest eingebaute Fileselectorbox, die sich nicht abschalten läßt. Ich könnte mir vorstellen, daß einige Anwender doch lieber ihre eigene Box benutzen würden. Zu guter Letzt sollten noch die Fehler abgestellt werden, die mit Turbo ST auftreten.

Insgesamt gesehen ist das TOS-Construction-Set ein ausgereiftes Programm, mit dem es problemlos möglich ist, den gnadenlos tristen Desktop-Symbolen und -Fehlern etwas auf die Sprünge zu helfen. Der Preis von DM 60,- ist nicht übertrieben und kann sich durchaus sehen lassen.

MP

Bezugsadresse:
Rosin Datentechnik
Peter-Spahn-Straße 4
W-6227 Oestrich-Winkel 2
Tel. (06723) 4978

Hard&Soft

Unser System setzt Maßstäbe

in Leistung, Zuverlässigkeit und Geschwindigkeit, ist kaum zu hören und natürlich vollkommen im Atari ST Design gehalten. Die ausgefeilte, grafische Benutzerführung macht die Bedienung spielend leicht.
... mit weniger Qualität sollten Sie sich nicht zufrieden geben.

Lieferbare Ausführungen

SCSI Ultra Speed Drive

* 52 MB · 17 ms · 64 KB Cache 1198,-
* 85 MB · 19 ms · 64 KB Cache 1498,-
* 105 MB · 17 ms · 64 KB Cache 1649,-
* 120 MB · 15 ms · 64 KB Cache 1898,-
* 170 MB · 15 ms · 64 KB Cache 2298,-
* 210 MB · 15 ms · 64 KB Cache 2548,-

SCSI Speed Drive

49 MB · 28 ms 1098,-
85 MB · 24 ms 1398,-

* Wechselplatte SCSI Speed Drive 44 .. 1398,-
Streamer SCSI Speed Drive 155 1798,-
Einbaufestplatte Mega ST
* SCSI Ultra Speed Drive 52 (LPS) 1049,-
* SCSI Ultra Speed Drive 105 (LPS) 1498,-

* **2 Jahre Garantie**

Vorbildlicher Service

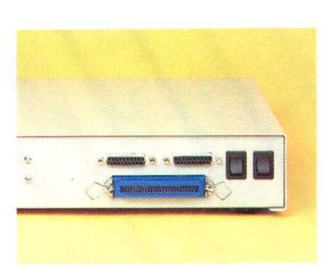
prompte Erledigung technischer Überprüfungen, Anpassungen und Reparaturen

Service Hotline

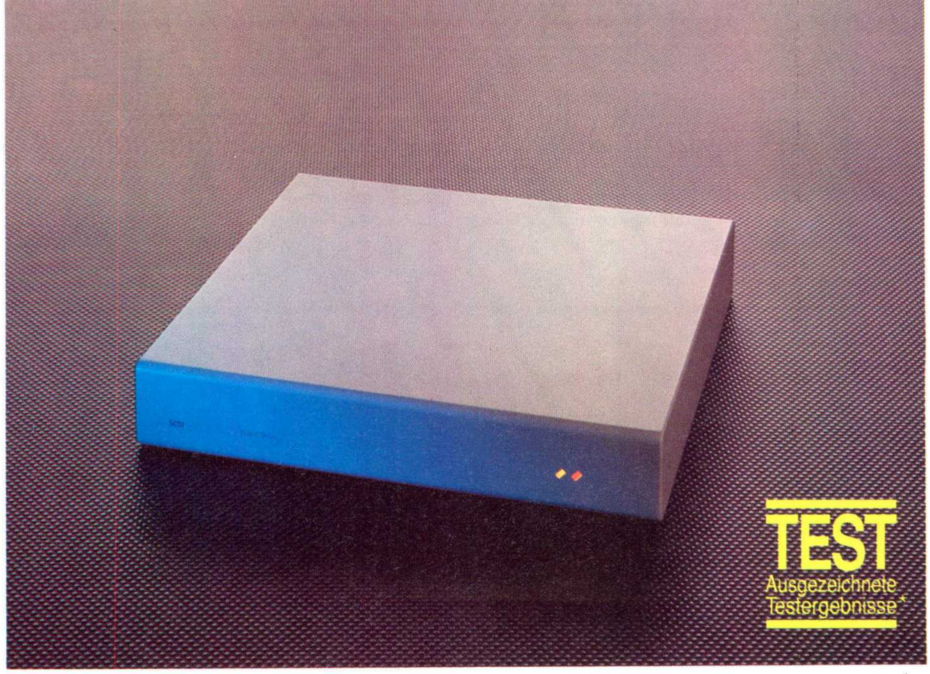
Im Falle eines Defektes innerhalb der Garantiezeit wird die Festplatte von unserem Zustelldienst United Parcel Service (UPS) bei Ihnen abgeholt. Die Kosten für diesen zusätzlichen Service tragen wir.

Rückgaberecht

per Versand gekaufte Platten können binnen 7 Tagen zurückgegeben werden.



SCSI Schnittstelle an Geräterückseite herausgeführt



* »... ein sehr gutes Preis/Leistungsverhältnis« (ST Computer 4/90)
»Fazit – ein Referenzmodell auf dem Festplattenmarkt« (ST Magazin 8/90)



Akzente
Softwarevertrieb
Adimens ST plus 314.–
GFA-Basic V 3.5 215.–
Aditalk ST plus 314.–
GFA-Assembler 129.–
Tempus Word 518.–
AT-Speed 429.–
Tempus V 2.10 108.–
Speed-Bridge II 48.–
Steckbare Erweiterung auf 2 MB 448.– auf 4 MB 698.–
SCSI Speed Drive 50, 28 ms, SCSI-Port 1348.–
SCSI Ultra Speed Drive 40, 19 ms, 64 KB Cache 1448.–
Preisliste kostenlos!
Akzente Softwarevertrieb
7080 Aalen · Postf. 1672 · Schlehenweg 12
Tel. (0 73 61) 3 66 06 · Fax (0 73 61) 3 66 07



KOLIBRI GRAFIK GMBH
ATELIER FÜR GRAPHIK & BESCHRIFTUNGSTECHNIK
45 OSNABRÜCK MÖSERSTR. 2
TEL 0541/ 22422
Wir schneiden Ihre
Vektorschriften & Graphiken
in CVG, GEM u. VEK - Format, aus
SELBSTKLEBEFOLIEN
Für den Privatgebrauch & für Beschriftungen von Schildern, Fahrzeugen u. Lichtwerbeanlagen
CUT-service
sprechen Sie mit uns !



DTP-Grundlagen

Teil 2 Visitenkarten & Briefbogen

Am Anfang war das Logo - oder doch zumindest die Idee desselben. Die Praxis wird Ihnen zeigen, daß gerade Ihre Kunden eine ganze Menge „konkreter“ Ideen von dem Logo haben, das Sie Ihnen entwerfen sollen. Auch wenn Sie denken, daß sich an dem Entwurf bestimmt nichts mehr ändern ließe: Ihr Auftraggeber wird Änderungswertes finden!

Aber er muß mit dem Entwurf, den Sie ihm ja verkaufen wollen, oft viele Jahre leben -und zwar zufrieden! Und er will Alternativen sehen, und aus diesen dann vielleicht wieder Ihre allererste Idee wählen. Jedoch ist es dann seine Entscheidung, und die sollten Sie ihm immer durch Entwurfsvarianten ermöglichen.

Die ersten Schritte

Ich will solch einen Gestaltungsablauf und die damit verbundenen Überlegungen einmal anhand eines Visitenkartenentwurfs darstellen. Wie Sie mit solch einem Entwurf im Calamus eine professionelle Druckvorlage erstellen können, mit der Ihr Drucker auch etwas anfangen kann, wird dann im nächsten Monat ausführlich behandelt (Bild 1).

Der Schriftzug *Subito* wurde nach der handschriftlichen Vorlage des Auftraggebers bearbeitet und gescannt. Es kommt gar nicht so selten vor, daß in die Neugestaltung einer Geschäftswerbung schon vorhandenes grafisches Material eingebunden werden muß. Ein Scanner ist hier also unerläßlich, jedoch reicht einer der handelsüblichen „Handyscanner“ für diese Zwecke völlig aus. Zur weiteren Bearbeitung nun zwei Arbeitsschritte, die Ihnen im weiteren Verlauf einiges an Streß ersparen können. Ist der Entwurf gelungen, werden Sie mit diesem unter Umständen eine ganze Geschäftspapierfamilie zu gestalten haben und zusätzlich (das ist jedenfalls meine Erfahrung) vielleicht auch noch KFZ-Aufkleber, Tüten, Prospekte usw. Im Falle *Subito* wurde aus dem eingescannten Schriftzug zuerst ein Mini-Vektor-Font erstellt, der nur aus den 6 Buchstaben besteht (Bild 2). Zu diesem Zweck benötigen Sie einen Fonteditor (DMC, Didot). Ich selbst arbeite mit Didot-“Lineart“, einer tollen Kombination aus Vektorgrafikprogramm und Font-Editor. Mit „LineArt“ haben Sie unter anderem die Möglichkeit, Vektorvorlagen zu erstellen und diese jederzeit als „CFN“ (Font) oder „CVG“ (Calamus-Vektorformat) zu exportieren, was für die weiteren Arbeiten eine erheblich größe Flexibilität und Zeitersparnis bedeutet. Mit dem fertigen Mini-Font können Sie dann im Calamus einfache Schriftvarianten erstellen, um spielerisch eine erste Entwurfsidee zu entwickeln. Ist der Entwurf „abgesegnet“, das heißt, hat der Auftraggeber sein OK für die weitere Gestaltung und den Druck gegeben, sollten Sie versuchen, das gesamte Logo soweit wie möglich als Vektorgrafik zu entwickeln. Auch wenn der Zeitaufwand anfangs etwas groß ist, haben Sie so das Logo für spätere Gestaltungen in unterschiedlichen Größen immer in den gleichen Proportionen vorliegen. Versuchen Sie das mal mit einem Gemisch aus Rasterbild und Vektor-Fonts!

Nicht unbedingt identisch

In unserem Beispiel war innerhalb anderer Subito-Werbemaßnahmen die Form eines gleichschenkligen Dreiecks schon vorgegeben. Es wäre nun aber sicher ganz falsch, zur Gestaltung der weiteren Arbeiten diese schon vorhandene Form einfach auf das neue Logo zu übertragen und nur noch mit den nötigen Zusätzen zu versehen - auch wenn dieses der in diesem Zusammenhang häufig anzutreffende Begriff *corporat identity* zu suggerieren scheint. Unter CI versteht man, einfach ausgedrückt, das einheitlich gestaltete Erscheinungsbild einer Firma in allen Bereichen: gleicher Firmenschriftzug, gleiches Layout in Broschüren und Pressemitteilungen, gleiche Farben usw. Die Farben einer Firma können in manchen Fällen sogar einen höhe-

ren Wiedererkennungswert haben als das Logo selbst. Sie sollten daher auf allen Gestaltungen vorhanden sein, auch dort, wo das Logo selbst gar nicht erscheint.

Die Vorgehensweise für die verschiedenen Subito-Gestaltungen bestand nun darin, die besonderen Merkmale, oder, ohne gleich Plato zu zitieren, die den Gestaltungsmitteln zugrundeliegende Idee aufzunehmen und sie auf das jeweils andere Medium anzuwenden. Für *Subito* waren dies: Liniendreieck, Schriftzug und Farbgebung.

Bild 1: Der erste Entwurf - frisch aus dem Laserdrucker



Bild 2: Ein Logo-Mini-Font, in Calamus etwas gedreht, ermöglicht das Durchspielen vieler Entwurfsvarianten.

Mit dem Logo auf die Karte

Die einheitliche Gestaltung besteht in diesem Fall also in der Verwendung eines „Liniendreiecks" und nicht unbedingt in dessen geometrischer Form (rechtwinklig, gleichseitig). Genau so, mit etwas Kreativität Ihrerseits, sollte *corporat identity* im Sinne eines einheitlichen Erscheinungsbildes auch verstanden werden. Gleiches gilt für die Verwendung der Farbe. Da das gesamte Logo in Weiß und Grau angelegt ist, kommt der Verwendung einer „echten" Farbe, in diesem Fall Gelb, sowieso eine herausragende Bedeutung zu. Die der Gestaltung zugrundeliegende Idee war es, Gelb als „grafisches Element" zu übernehmen, und nicht, entgegen mancher Gestaltungsregeln, die Verwendung dieser Farbe auf „i-Punkt" und „T-Querstrich" (so die Farbgebung der Fensterbeschriftung) auch für die weiteren Arbeiten festzulegen. Man sollte auch daran denken, daß z.B. ein Ladenfenster nicht nur ein ganz anderes Format zur Verfügung stellt als eine Visitenkarte, sondern auch mit einem völlig anderen Zweck verbunden ist und somit eine entsprechend andere Wirkung auf den Betrachter ausüben soll.

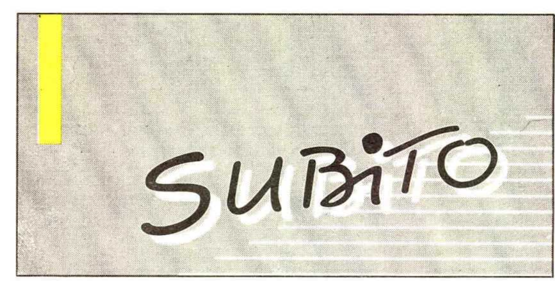


Bild 3: Die Gestaltung der Visitenkarten-Vorderseite...

Bild 4: ...und der Rückseite

Für die Visitenkartengestaltung wurde aus diesen Gründen das gelbe Rechteck in die linke obere Ecke der Visitenkarte positioniert. Das Resultat ist eine ausgewogene Aufteilung der zu getaltenden Fläche, wobei durch das Absetzen der beiden Elemente voneinander auch noch ein kleiner Bruch in der Gestaltung bewirkt wird. Reine Ausgewogenheit unter homogenen Gestaltungselementen wirkt langweilig! Der Pfiff oder das „Besondere Etwas" , das manche Gestaltungen auszeichnet, wird oft nur durch solch eine Auflösung der Gleichmäßigkeit bewirkt (Bild 3).

Auf der Rückseite der Karte kann auf den Schriftzug *Subito* verzichtet werden. Das Gleichgewicht wird durch die Positionierung des Adressenfeldes auf die linke Seite unten hergestellt. Alle übrigen Elemente sind vorhanden: Liniendreick, gelbes Rechteck, Schriftzug (Bild 4).

Grundsätzlich sollten Sie vor Beginn einer Visitenkartengestaltung genau wissen, in welchen Bereichen diese vorwiegend genutzt werden soll. Als reine „Besuchskarte", auf der lediglich Name, Anschrift und Telefonnummer vermerkt sind, wie es noch bis vor wenigen Jahren üblich war, läßt sie sich selbst im privaten Bereich kaum noch finden. Anwendung findet sie zum Beispiel als Werbekarte, zur Leistungsübersicht einer Firma, zur reinen Selbstdarstellung usw. Zur Gestaltung werden die wichtigsten Elemente des Briefbogens übernommen: Logo, Name/Funktion des Mitarbeiters, Farbgestaltung, typographische Stilmittel. Auch aufgrund des kleinen Formats sollten Sie auf der Visitenkarte nur wenig Schriften in unterschiedlichen Größen verwenden.

Als Standardformat für die Größe einer Visitenkarte galt ursprünglich DIN A7 (halbe Postkarte). Dieses Format werden Sie aber kaum noch antreffen; es ist einfach zu groß. Heute gilt als Maßstab die Größe einer Kreditkarte, da die für sie vorgesehenen Plazierungen (z.B. in Brieftaschen, Scheckkarten und Taschenrechnern) sich dann auch für Visitenkarten nutzen lassen.

Ein Blatt für viele Zwecke: Briefbogen

Wenn Sie bei der Gestaltung einer Visitenkarte noch relativ frei Ihrer Kreativität nachgehen können, müssen Sie bei der Anlage eines Briefbogens einige Normen beachten (Bild 5), und sei es auch nur, damit eine Adresse auch nachher im Fenster des Langumschlags noch zu sehen ist. Das Firmenlogo selbst findet sich in der Mehrheit aller Briefbogengestaltungen rechts oben in der Kopfleiste eines DIN A4-Bogens. Diese Plazierung erleichtert auch ein Wiederfinden beim Durchblättern im Ordner. Für die Schrift einer Briefbogengestaltung gilt das Gleiche wie für die Farbe: Die von Ihnen gewählten typographischen Mittel gehören wie das Logo zum äußeren Erscheinungsbild der Firma und sollten auf allen Gestaltungen Anwendung finden. Aufgrund der mitunter recht teuren Mehrfarbdrucke werden in vielen mittelständischen Unternehmen farbig angelegte Briefbogen auch für den Formulargebrauch benutzt. Auch Sie selbst können mit dieser Methode Eigenkosten sparen, indem Sie in Ihrem Calamus den Briefbogen gleich so anlegen, daß der schwarz zu druckende Text erst via Laserdrucker mitgedruckt wird. Sie speichern für diesen Vorgang die für den Schwarzdruck vorgesehene Seite als einzelnes CDK-Dokument. Wenn Sie nun ein Formular anlegen, z.B. eine Rechnung schreiben müssen, wird dieses Leerformular geladen und wie gewohnt ausgefüllt. Danach speichern Sie es mit *Datei/sichern unter...* unter einem neuen Namen, zum Beispiel der Rechnungsnummer, ab. Mit diesem Verfahren haben Sie das Leerformular immer auch leer zur Verfügung.

Die weiteren Informationsfelder des Briefbogens (Telefon, Adressen, Bankverbindungen) können Sie beliebig plazieren. Üblich ist hier: unter das Logo oben rechts oder in die Fußleiste. Für den Zahlensatz von z.B. Telefon und Bankverbindung gibt es aber dennoch einige Regeln, die Sie aus Gründen der besseren Lesbarkeit beachten sollten.

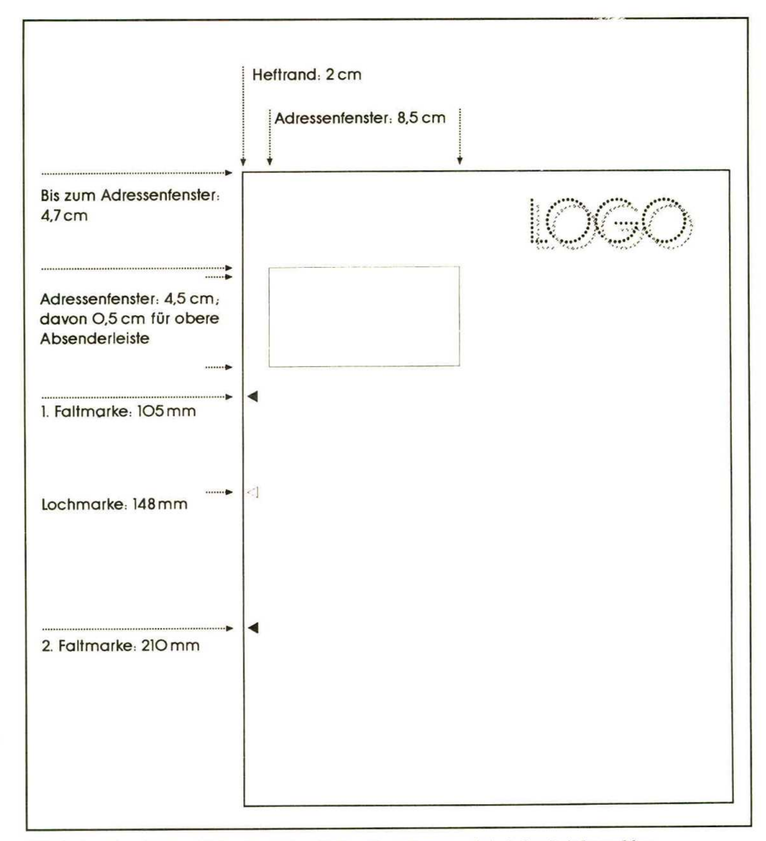


Bild 5: Auf das richtige Maß gebracht, paßt Ihre Gestaltung auch in jeden Briefumschlag.

Telefonnummern: werden bei mehr als 3 Ziffern, von hinten angefangen, in 2er-Gruppen unterteilt. Bankkonten: von hinten in 3er-Gruppen. Bankleitzahl (hat immer 8 Stellen): von links in 3er-Gruppen, am Schluß eine 2er-Gruppe. Postfach: von hinten immer in 2er-Gruppen.

Jetzt aber Schluß mit den Formalitäten! Im nächsten Monat werden wir uns nach Typographie und Satz endlich mit der Bearbeitung unserer Dokumente für den Druck beschäftigen: Das Aufbereiten der Dokumente für die verschiedenen Druckverfahren, Nutzenerstellung für Aufkleber und Visitenkarten, Vorbereitung zur Filmbelichtung, DTP-relevante Unterschiede im Sieb- und Offset-Druck; mit dem nötigen Wissen zur Druckvorlagenerstellung lassen sich nicht nur Frust, sondern auch eine Menge unnötiger Kosten sparen.

Ich möchte Sie auch jetzt noch einmal bitten, mir Ihre Fragen und Anregungen zukommen zu lassen. Im 4. und letzten Teil dieser DTP-Grundlagen werde ich auf Anwenderfragen und -probleme aus der DTP-Praxis eingehen - also ran an die Tasten!

Jürgen Funcke



2211

Philatelie verwaltet Briefmarkensammlungen. Zeigt Umfang und Wert der Sammlung anhand der gültigen Katalogpreise. Das Programm listet Einzelmarken und Sätze (s/w, MB, S).

Tribu 1.5 ist in dieser Version auch zur Verwaltung von Aktiendepots geeignet (s/w).



Karteye 3.0 verwaltet Daten aller Art auf professionelle Weise. Makrofunktionen, Maskeneditor, Serienbriefe und zahlreiche Datenmasken werden mitgeliefert (s/w).

Software 1.0 dient der Eingabe und Verwaltung von Diskinhalten. Bei 1MB Speicher werden ca. 10.000 Einträge erfaßt. Gute Gestaltung und einfache Bedienung zeichnen das Programm aus (s/w).

Cassette 3.8 bedruckt Casettenhüllen auf einfache Weise. Viele Programmfehler der Version 3.7 sind jetzt behoben, die Anpassung an beliebige Drucker ist möglich (ST/TT, s/w).

2212

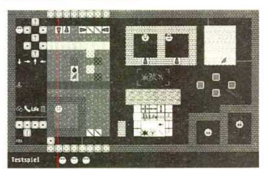


Das Fußball Studio verwaltet Tabellen der Fußball-Bundesliga auf durchdachte und vielseitige Weise (s/w).

Sporttabelle enthält die 1. und 2. Bundesliga, Saison 89/90 (s/w).

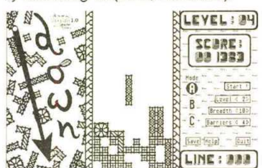
Uni-Liga 1.6 dient der Verwaltung sämtlicher Sportligen, sofern keine Kommastellen in Ergebnissen verwendet werden (s/w).

2213



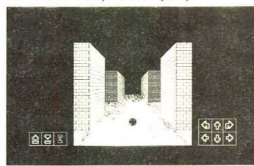
Eiskalt ist ein Actionspiel mit 100 Ebenen, von denen die ersten 11 frei spielbar sind (s/w, J, S).

Newtris 2.0 verbindet Tetris und Columns in einem gemeinsamen Programm, das auch als Accessory lauffähig ist (ST/TT, PRG/ACC).



down 1.0c spielt Tetris mit erweitertem Funktionssatz, 4 bis 20 Felder Breite, drei Varianten und einer interessanten Gestaltung (s/w).

Pack den Sack macht den Spieler zum Millionär, indem er Geldsäcke einsammelt und in einer Tonne deponiert (s/w).



A Maze zeigt ein sehr schön gestaltetes 3D-Labyrinth, in dessen Gängen der Spieler nach Bällen suchen soll. Unkomplizierte Mausbedienung (s/w).

Catch'Em 1.4 ist voller Würmer, die sich über den Bildschirm bewegen und vom Spieler eingefangen werden sollen (ST/TT).

2214



Giro-STB 1.9a erledigt den beleglosen Zahlungsverkehr, samt Datenübernahme aus Fibu und Kalkulationsprogrammen. Paßwort, Wildcard-Suche und viele weitere Funktionen (Platte oder 2 Floppies).

Ask Me 2.0 ist ein hierarchiches Informationssystem für Faktenwissen aller Art: Vokabeln, Schallplatten, Daten und vieles mehr. Ask Me verarbeitet Bilder und Blöcke, es eignet sich daher auch als Lernsystem und Grafikabenteuer.

2215

Ostern zeigt an, auf welches Datum der Ostersonntag fällt, von 1700 bis 2199.

Konditionstest 1.1 ermittelt die derzeitige Kondition anhand von Pulsmessungen vor und nach körperlicher Betätigung (s/w).



AutoKost 2.5 verwaltet Termine und Kosten die bei Betrieb und Wartung Ihres Fahrzeugs anfallen (ST/TT, s/w)



AUA beinhaltet eine Heilkräuter-Datenbank, die wirksame Gegenmittel zu den verschiedensten Beschwerden bereithält (s/w).

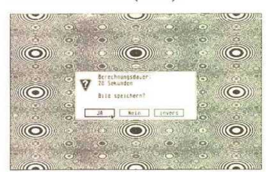
Konto 2.1 verwaltet Daueraufträge, Soll und Haben Buchungen von Giro und Sparkonten (s/w).

Astro 1.3 enthält alle Funktionen zur Herstellung eines Radix Horoskops (s/w).

2216



Hüpfer 4.0 berechnet Grafiken aufgrund der einzugebenden Werte a, b und c. Algorithmus aus Spektrum der Wissenschaft, viele neue Funktionen (ST/TT).



Chaos-Studio 1.0 berechnet viele herrliche Bilder: Pickover Mengen, Apfelmännchen, Psycho-Bilder, Mandalas usw. Einfache Bedienung, ohne Parametereingaben. Hohe Geschwindigkeit (s/w).

Drachen 1.1 zeichnet Fraktale der Dimension 1 und 2, speichert und druckt sie bei Bedarf als Poster aus (s/w).

2217

Funktionsinterpreter 1.3a überprüft die korrekte Eingabe von Funktionen und berechnet deren Wertetabellen (G).

Def2Lst 1.4 wandelt DEF-Dateien des KRSC 2.2 in LST-Files für GFA-Basic 3.XX (ST/TT).

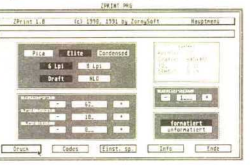
B-Shell 1.8 vereinfacht den Umgang mit GFA-Basic und diversen Hilfsprogrammen.

ENSONIQ dient dazu, ENSONIQ-MIRAGE-SOUNDDISKETTEN mit angenehmen Aufklebern zu versehen. Druckt auf LC10, NEC P6 und Kompatiblen (s/w).

ASMxxx 1.1 ist ein Assembler, der in speziellen Version unter GEMDOS und DOS eingesetzt werden kann. Gute Dokumentation aller Programmteile (ST/TT/DOS).



Filecomp 2.2 prüft beliebige Dateien auf Unterschiede und erlaubt das Editieren dieser Datein in begrenztem Umfang (ST/TT, s/w).



ZPrint 1.8 druckt ASCII Textdateien auf sämtlichen Druckern. Läuft in dieser Version als PRG und Accessory (ST/TT/KAOSDESK).

WakeUp ruft den Laserdrucker ins Leben zurück, wenn er nach dem Zuschalten nicht drucken will.

2218



HP-Desk Hardcopytreiber für den HP-Deskjet.

Notiz 2.9 nimmt Notizen und Ideen auf, speichert und druckt sie bei Bedarf aus (ST/TT).

RamFrei 3.2 zeigt den freien Speicher an. ST- und TT-RAM werden getrennt dargestellt (ST/TT).

Super Boot 6.0 lädt AUTO Programme, Accessories, GEM Anwendungen und Begrüßungsbildschirme – im Degas, NEO oder TNY Format – beim Systemstart. Das Programm bietet Paßwort Schutz gegen unerlaubte Zugriffe, arbeitet mit dem Switcher "Revolver" zusammen und läßt sich mit dem beiliegenden Construction Set sehr leicht einstellen.

STRESOR schließt den ST ab

Besonderheiten:

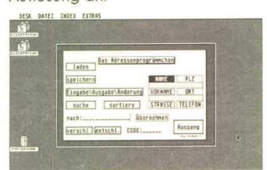
f	= Läuft nur in Farbe
s/w	= Läuft nur monochrom
E90	= Läuft auch mit Emula 5.1 von Diskette 2090.
e	= in englischer Sprache
J	= Joystick notwendig
MB	= 1MB RAM erforderlich
S	= Shareware
G	= GFA-Quellcode liegt bei
K	= Kontaktkarte eingebaut

und läßt nur den Inhaber des Paßworts an sensible Daten und Programme (s/w).

Sokrates 1.0 belegt die Tastatur mit Altgriechisch, damit viele andere Programme diese Sprache schreiben und verarbeiten können. Kompatibel zu vielen Text und Lernprogrammen.

MstrMind 1.0 spielt als Programm oder Accessory auf Atari ST und TT.

RDDI paßt die Einstellung des Desktops optimal der gewählten Auflösung an.



Das **Adressenprogrämmchen** ist ein kompakt gestaltetes Accessory, das seine Adressdaten an Textprogramme übergibt, Daten ver und entschlüsselt.

SideKick ist das Taschenmesser für den TT. Viele nützliche Funktionen warten im Hintergrund (G, TT).

SCRSAV-H schont den TT-Monitor TTM194, indem es ihn nach gewisser Zeit dunkel schaltet (G, TT).

2219



Sternzeit berechnet den Sternenhimmel anhand der Daten von ca. 300 Sternen, mit 1/10 Minute Genauigkeit (G, s/w).

Die Erde ermittelt Tag und Nachthälften der Erde, zeigt Sonnenauf und untergang an (f, s/w & Großmonitor).

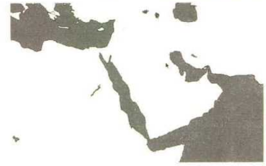




World 1 enthält über 100.000 Erdkoordinaten, mit einer atemberaubenden Auflösung. Disk 2220 wird benötigt (f, s/w, G).

2220

World 2 enthält Grönland, Nord- und Süd-Amerika. World 2 ist nur mit Disk 2219 verwendbar.



Mondkugel zeigt die Oberfläche des Mondes in dreidimensionaler Darstellung und Rotation, inklusive Mondrückseite (f, s/w, Großmoni).

Ritzel 1.0 berechnet Übersetzungsverhältnisse einer Fahrrad-Kettenschaltung und stellt sie grafisch dar. Ritzelkombinationen können optimiert werden (s/w).

PD-Szene schon gelesen?

Jetzt mit 84 Seiten und 10.000 Auflage. Bei Ihrem Pool-Händler.

PD-Pool sucht noch einige gute Programme zur Veröffentlichung auf den Disketten 2221 - 2230. Die Vorstellung erfolgt gleichzeitig in mehreren großen ST- und PD-Zeitschriften.

Die vorgestellten Disketten erhalten Sie exklusiv bei folgenden PD-Anbietern:

B.I.T.S.
Jagowstraße 17
1000 Berlin 21
030 / 3938203

HD-Computertechnik
Pankstr. 61
1000 Berlin 65
030 / 4657028-29

M.Damme -Druck&Computer
Grambeker Weg 40
W-2410 Mölln
04542 / 87258 (ab 16 Uhr)

T.U.M.-Soft&Hardware
Hauptstr. 67
2905 Edewecht
04405 / 6809

3 ½ SOFTWARE
Wendenstr. 45
3300 Braunschweig
0531 / 13624

INTASOFT
Nohlstr. 76
4200 Oberhausen 1
0208 / 809014

H. Richter Distributor
Hagenerstr. 65
5820 Gevelsberg
02332 / 2706

Eickmann Computer
In der Römerstadt 249
6000 Frankfurt 90
069 / 763409

IDL Software
Lagerstraße 11
6100 Darmstadt 13
06151 / 58912

Weeske Computer
Potsdamer Ring 10
7150 Backnang
07191 / 1528-29 od. 60076

Wacker GmbH
Bachstraße 39
7500 Karlsruhe 21
0721 / 554471

=PD-Express= J. Rangnow
Ittlinger Straße 45
7519 Eppingen-Richen
07262 / 5131 (ab 17 Uhr)

Duffner Computer
Habsburgerstr. 43
7800 Freiburg
0761 / 56433

Simonis Computersysteme
Ubostraße 55
8000 München 60
089 / 8632609 (ab 18 Uhr)

Schick EDV-Systeme
Hauptstraße 32a
8542 Roth
09171 / 5058-59

PDST - Michael TWRDY
Kegelgasse 40/1/20 / PF 24
A-1035 Wien
0222 / 75-27-212

GUBLER-COMPUTERS
Spalenring 75
CH-4009 Basel
061 / 239871

Die Zusammenstellung der PD-Pool 2000er Serie unterliegt dem Investitionsschutz: Die Serie darf in der vorgestellten Form nur von den hier angegebenen Pool-Teilnehmern verkauft werden. Im Interesse der PD-Programmierer beachten Sie bitte auch die Nutzungsbestimmungen der verwendeten PD und Shareware.

Wir suchen noch Fachhändler und PD Anbieter, die sich an Anzeigen und Messen beteiligen möchten. Tel.: 06151 / 58912, Herrn Schultheis verlangen.

☐ Scheck über DM ______ liegt bei, ich erhalte die Ware verpackungs- und versandkostenfrei (Ausland: Bitte Euroscheck in der Landeswährung des Händlers).

☐ Per Nachnahme. Nur Inland! (zuzüglich DM 6, Nachnahmegebühr).

☐ Bitte senden Sie mir die aktuelle Ausgabe der PD Szene, mit Komplettkatalog der 2000er Serie und ***Top Tausend*** PD Liste. DM 2,50 liegen bei.

2151	2161	2171	2181	2191	2201	2211
2152	2162	2172	2182	2192	2202	2212
2153	2163	2173	2183	2193	2203	2213
2154	2164	2174	2184	2194	2204	2214
2155	2165	2175	2185	2195	2205	2215
2156	2166	2176	2186	2196	2206	2216
2157	2167	2177	2187	2197	2207	2217
2158	2168	2178	2188	2198	2208	2218
2159	2169	2179	2189	2199	2209	2219
2160	2170	2180	2190	2200	2210	2220

DM 2,50
P.D. SZENE
News
Tests
Top Tausend
P.D. Challenge

Diskpreis: DM 8,- *

öS 60,- * / sFr 8,- *

* unverbindlich empfohlener Verkaufspreis

Lieferung an meine Adresse:
(Die neu vorgestellten Disketten 2211-2220 sind ab 15.04.91 lieferbar.)

____

____

____

____

STC 4/91

Gewünschte Disketten ankreuzen und Bestellschein an einen der oben angegebenen Anbieter einsenden.

Einkaufsführer

Hier finden Sie Ihren Atari Fachhändler

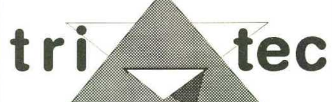
1000 Berlin

COMPUTERSYSTEME
Schlichting
...die etwas andere Computerei
COMPUTERSYSTEME, PLAYSOFTSTUDIO SCHLICHTING
COMPUTER - SOFTWARE - VERSAND GMBH & CO. KG
ATARI-FACHMARKT
MS-DOS FACHMARKT • NEC FACHHANDEL
Katzbachstraße 8 · 1000 Berlin 61
030/786 10 96



alpha computers g.m.b.h.
u. a. alphatronic, atari, commodore, dai, epson, sord mit pips, nec
hard-/software nach maß — servicetechnik
Kurfürstendamm 121a, 1000 Berlin 31 (Halensee)
Telefon 030/891 1082

Mangoldt – Weidlich – Co GbR



Computer – DTP – Sat-Antennen
Straßmannstraße 46, O – 1034 Berlin
Tel.: 6 74 50 56 (10⁰⁰ – 16⁰⁰)
Anrufbeantworter: 5 88 97 49

1000 Berlin



PD SERVICE — **Je Disk 5,- DM**

Über 5000 PD-Disketten verfügbar!
z.B. ACS, AMIGA JUICE, AMIGALIBDISK, ANTARES, AUGE 4000, AUSTRIA, BAVARIA, BEL AMI, BRUNOSOFT, CACTUS, CHEMIE, CHIRON, ES-PD, FAUG, FRANZ, GERMAN, GET IT, KIRSCHBAUM, MIDI-PD, OASE, OLLIS GAMES, PANORAMA, PAXUM, PFALZ, PUDOMIX, R-H-S, RPD (=CAM), 6 2 6, SCHATZ, S-DREAMS, TAIFUN, TAURUS, T.B.A.G., TIGER, TIME, TORNADOS, U.G.A., u. a.
und natürlich sämtliche KICKSTART-PD's!

Speichererweiterungen
Diskettenlaufwerke
Festplatten & Turbokarten
Anwendungssoftware
Disketten Großhandel
Fachliteratur & Zubehör
Desktop Video/Publishing
Reparatur Service

HD COMPUTERTECHNIK oHG
Pankstraße 61
1000 Berlin 65
Tel.:030/465 70 28

REPARATUR WERKSTATT
1000 Berlin 65 - Pankstr. 42

SERVICE STATIONEN
1/44, Lahnstr. 94, Tel:684 48 31
1/20, Schönwalder Str. 65, Tel:375 60 13

COMPUTERSYSTEME
Schlichting
...die etwas andere Computerei
COMPUTERSYSTEME, PLAYSOFTSTUDIO SCHLICHTING
COMPUTER - SOFTWARE - VERSAND GMBH & CO. KG
ATARI-FACHMARKT
MS-DOS FACHMARKT • NEC FACHHANDEL
Mönchstraße 8 · 1000 Berlin 20
030/786 10 96

1000 Berlin

DATAPLAY
Bundesallee 25 · 1000 Berlin 31
Telefon: 030/861 91 61



COMPUTERSYSTEME
Schlichting
...die etwas andere Computerei
COMPUTERSYSTEME, PLAYSOFTSTUDIO SCHLICHTING
COMPUTER - SOFTWARE - VERSAND GMBH & CO. KG
ATARI-FACHMARKT
MS-DOS FACHMARKT • NEC FACHHANDEL
Wilh.Pieck-Straße 56 · Potsdam
030/786 10 96



PLAYSOFT-STUDIO
Schlichting
...die etwas andere Spielerei
PLAYSOFTSTUDIO SCHLICHTING
COMPUTER - SOFTWARE - VERSAND GMBH
BERLINS NEUE DIMENSION FÜR COMPUTERSPIELE
Katzbachstraße 8 · 1000 Berlin 61
030/786 10 96

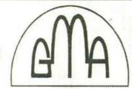
2000 Hamburg

Planen • Beraten • Realisieren



ATARI Fachhändler
Münsterstraße 9, D-2000 Hamburg 54
Telefon 040/56 19 09-0
Telefax 040/56 19 09-80

G.M.A.
Wandsbeker Chaussee 58
2000 Hamburg 76



Einziges, autorisiertes Atari-DTP-Center in Hamburg.
Telefon: 040 / 25 12 41 5-7

DTP u. Werbeservice
Soft- und Hardwareversand
Viola Jaap
Berner Heerweg 512
2000 Hamburg 72
Telefon (0 40) 6 44 06 68

2000 Norderstedt



Ulzburger Str. 2
2000 Norderstedt
Tel. 040 / 527 30 47

2120 Lüneburg

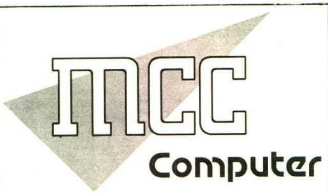
Sienknecht
Bürokommunikation
Beratung - Verkauf - Werkstatt
Heiligengeiststr. 20, 2120 Lüneburg
Tel. 04131/46122, Btx 402422
Mo.-Fr. 9⁰⁰-18⁰⁰ und Sa. 9⁰⁰-13⁰⁰

2210 Itzehoe

Der Computerladen
Inhaber Ulrich Bubel Martin Kopplow

Coriansberg 2 · 2210 Itzehoe
Telefon (04821) 3390/91

2300 Kiel



MCC Computer GmbH Holzkoppelweg 19a
2300 Kiel 1 TEL 0431/54381 FAX 541717

2800 Bremen

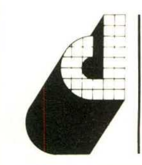


Faulenstraße 48—52
2800 Bremen 1
Telefon (0421) 170577

2940 Wilhelmshaven

Radio Tiemann
ATARI-Systemfachhändler
Markstr. 52
2940 Wilhelmshaven
Telefon 04421-26145

3000 Hannover



DATALOGIC COMPUTERSYSTEME
ATARI ST- COMPUTER HARDWARE SOFTWARE
BERATUNG SERVICE VERKAUF
CALENBERGER STR. 26
3000 HANNOVER 1
TEL.: 0511 - 32 64 89

3000 Hannover

COM DATA
Am Schiffgraben 19 · 3000 Hannover 1
Telefon 0511-326736



HD COMPUTER-TECHNIK oHG
HANNOVER
Public Domain Service
Hardware & Zubehör
Fachliteratur
Hildesheimer Str. 118
3000 Hannover 1
Tel.: 0511/809 44 84

3400 Göttingen

Büroeinrichtungs-Zentrum

Wiederholdt
3400 Göttingen-Weende
Wagenstieg 14 – Tel. 0551/3857-0

4000 Düsseldorf

HOCO
EDV ANLAGEN GMBH
Ellerstraße 155
4000 Düsseldorf 1
Telefon 0211/785213

Hard und Software
Werner Wohlfahrtstätter

Atari	**Ladenlokal**
Public Domain	Irenenstraße 76c
Atari Spiele	4000 Düsseldorf-Unterrath
Atari Anwender	Telefon (0211) 429876

BERNSHAUS GmbH
Bürotechnik — Bürobedarf
Cäcilienstraße 2
4000 Düsseldorf 13 (Benrath)
Telefon 0211-719181

4010 Hilden

4010 Hilden
Fax: 02103/31820
Industrieberatung
Feldstr. 2 – Tel.: 02103/5927-9
Ladenlokal
Gustav-Mahler-Str. 42-44 · 02103/31880
Versand
Regerstr. 34 – Tel.: 02103/41226

Weide ELEKTRONIK

DTP - Center
CAD - Studio
Satzbelichtungen
Schneidplottersysteme

Wir führen Satzbelichter, Entwicklungseinheiten, Großbildschirme, Scanner, (Schneid-) Plotter und Zubehör, Optische Platten, Fest- und Wechselplatten, Streamer, OCR-Software.
Wir vernetzen Ihre Computer, auch MS-DOS mit ATARI ST/TT.
Wir unterhalten eine eigene **Reparaturwerkstatt**.
Unser **Außendienst** berät Sie gerne auch in Ihren Räumlichkeiten.
Wir sind Atari DTP Center, Dupont und Graphtec Vertriebspartner, **Linotype Business Center**, Olivetti Systemspartner.

4150 Krefeld

NEERVOORT | EDV
Kommanditgesellschaft
Nordwall 96
D-4150 Krefeld 1
Tel.: 02151-772056
FAX: 02151-770995
BTX: 02151772056

Computer- Hard- & Software
Ein- & Mehrplatzsysteme
Komplettlösungen

4200 Oberhausen

Redakteur · **Spectre 128**
dBMAN für TOS, DOS, UNIX, MAC, AMIGA und andere Systeme
by COMPUTER MAI

dBMAN-Vertretung für NRW und BENELUX
ISYS-COMPUTER GbR
Tel.: 0208/655031 · Telefax: 0208/650981
Max-Eyth-Straße 47 · 4200 Oberhausen 11

4320 Hattingen

Ihre Tür zur Zukunft:


Hattingen, Große Weilstr. 18-20, Telefon (02324) 20973

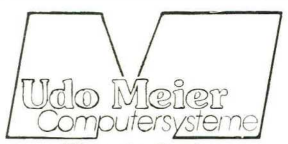
4430 Steinfurt

CBS GmbH
COMPUTERSYSTEME
Tecklenburger Str. 27
4430 Steinfurt-Burgsteinfurt
☎ 02551/2555

4500 Osnabrück

Heinicke-Electronic
Meller Str. 43 • 4500 Osnabrück
Fax (0541) 58 66 14
Telefon (0541) 58 76 66
Wir liefern Micro-Computer seit 1978

7700 Singen



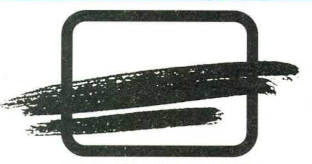
Ringstraße 4
Telefon (077 31) 682 22

7750 Konstanz

neu: ATARI Desk Top Publishing-Center



7800 Freiburg



DUFFNER COMPUTER
Habsburgerstraße 43
7800 Freiburg
Telefon 0761/56433
Fax 0761/551724

7890 Waldshut-Tiengen



service gmbh
Lenzburger Straße 4
7890 Waldshut-Tiengen
Telefon 077 51 / 30 94

7918 Illertissen

bidtech gmbh
technische Informationssysteme
Computerladen

Marktplatz 13
7918 Illertissen
073 03/50 45

8000 München



COMPUTER + BÜROTECHNIK
COMPUTER · SOFTWARE · PERIPHERIE
BERATUNG · TECHN. KUNDENDIENST

INGOLSTÄDTER STRASSE 62L
EURO INDUSTRIE-PARK • 8000 MÜNCHEN 45
TELEFON 089 / 31 81 95-0 • TELETEX 89 83 41

City Studio
Rindermarkt 6, 8000 München 2
Tel. 089 / 2 60 98 01

8000 München

schulz computer

Schillerstraße 22
8000 München 2
Telefon (0 89) 59 73 39

Beratung · Verkauf · Kundendienst

8032 Gräfelfing



Pasinger Straße 94 8032 Gräfelfing
Telefon 089 - 8548823
Fax 089 - 8541764

8150 Holzkirchen



Service und Beratung sind bei uns Inklusive

8200 Rosenheim



COMPUTER + BÜROTECHNIK
Kufsteiner Str. 11, 8200 Rosenheim
Tel. (08031) 38 00 30
Fax (08031) 1 53 34

8400 Regensburg

WITTICH
COMPUTER GMBH
Luitpoldstr. 2
8400 Regensburg

Tel. (0941) 562530
Fax (0941) 562510



8423 Abensberg

WITTICH
COMPUTER GMBH
Tulpenstr. 16
8423 Abensberg
☎ 0 94 43 / 4 53



8500 Nürnberg



8520 Erlangen

Computerservice Decker
Meisenweg 29 - 8520 Erlangen
Telefon 0 91 31 / 4 20 76

Bei uns werben bringt

GEWINN



Sprechen Sie mit uns.
Heim Verlag 0 61 51 / 56057

BUF

8700 Würzburg



Hardware · Software
Service · Schulung
computer center
am Dominikanerplatz
Ruf (09 31) 3 08 08 - 0

8720 Schweinfurt

Uhlenhuth GmbH
Computer + Unterhaltungselektronik
Albrecht-Dürer-Platz 2
8720 Schweinfurt
Telefon 09721 / 652154

8763 Klingenberg



Calamus-Belichtungen in Repro-Qualität auf Linotronic 300

Machen Sie den Preisvergleich !!!

Satz & Reprotechnik Hohmann · 8763 Klingenberg
Wilhelmstr. 7 · Tel. 09372/20750 · Fax 09372/20887



Kleinanzeigen

Biete Software

ADIMENS Anwendungen:
Standardlösungen und maßgeschneiderte Programme für Privat und Geschäft, Info unter 0221/7407194

GFA-Entwicklungspaket 3.0
ST Pascal plus 2.0, beides Originale mit Handbuch, je 100 DM, 0221/852262

Verkaufe wg. Systemwechsel ADI-Talk(plus), Beckertext 2.0, div. Omikron Lib's + Compiler und alles Originale; kompl. Sammlung Zeitschriften; 07841/25386

Verschiedenes

An Programmierer u. Softwarehäuser!
Sachbuchautor (Profi) schreibt oder korrigiert Ihre Handbucher.
R.D.Zens 02266/44228

Atari ST läuft nicht mehr? Ich repariere m. Garantie oder kaufe. Tel. 06151/24012

Atari ST + MS-DOS Computerclub Programme, Zeitung, Kurse, Tests u.v.m; kostenlose Info gegen frankiert. Rückumschlag, Demo-Diskette 5 DM: ASTUC, Kruppstraße 19, W-6290 Weilburg

---

Einsteiger/in! Programmieren auf Atari ST leicht gemacht m. lauffäh. Prozeduren und funkt. Omi-BASIC auf 3,5" nur 10 DM (Schein): A. Leung, St-Ulrich-Str. 2, 8064 Altmünster

---

Hilfe! Ich brauche dringend Tips zum Adventure Dragonflight. Wer hat es gelöst? Ich zahle bis zu DM 20,- für Komplettlösung und rufe zuruck.
Dietrich Feist. 06221/163784

ST-Magazin 7/88–1/91, ST-Computer 3/88–8/90, Happy-Computer 12/85–3/88, div. c't Atari-Magazin, PD, Complife usw à 3,–, Sonderhefte 6,–, Atari XL intern 20,–, 0731/35950

ST-Computer 2/89–12/90, Chip Hefte 1/87–12/90, ST-Magazin 1/89–12/90: nur komplett für 250 DM zu verkaufen
Tel. 02630/8198

ST-Computer, vollständige Sammlung von 6/87 bis 6/89; komplett oder jahrgangsweise für je 40,– zu verkaufen.
Tel. 089/7855365

Ein Anruf sagt mehr als 1000 Worte: BBS-FERWI, 0611/721704,
8N1, 300-2400 Bps

Selbsterstellte Dokumente und Grafiken zur Einbindung in Calamus (z.B.: Visitenkarten). Bitte melden bei: Michael Unter, Postfach 602226, 2000 Hamburg 60

Wer kann mir die Bauanleitung für den STAD-Scanner besorgen oder kopieren? Ulrich, Tel.: 07142/66601

ST-Computer, 1/88–12/90 für 120 DM zu verkaufen. Tel. 02222/63244

Antic-Magazin, orig. US-Zeitschrift für 8-Bit-Atari; Sammlung 1982–84; komplett oder jahrgangsweise zu verkaufen.
Tel. 089/7855365

---

2 Adimens-User im Ruhrgebiet gesucht, die gemeinsam mit mir ein Update von Adimens 2.x auf 3.1 fur 300,- DM (= 3 Updates) anstatt 450,- in einer Sammelbestellung kaufen.
Tel. 0211/782523, 18.30–24.00 Uhr.

ST-Computer-Hefte 1/87 bis heute, DM 150,–. Tel. 02306/13792

Tausche Top-Software! Liste mit Briefmarke an: Daniel Ehrat, Tosstalstr. 131, CH-8400 Winterthur

Suche Software

Atari-DR-Entwicklungspaket gesucht.
Tel. 06181/45361

Jahresinhaltsverzeichnisse der ST-Computer auf Disk gesucht. Tel. 07666/5049

Multisync auch defekt für ST Betrieb, AD-DA Wandler für Sounddigitalisierung, suche außerdem: Spectrum 512, Cyberpaint und -sculpt.
Tel. 0731/35950, ab 16 Uhr

Suche PD-Version des Simulationsprogrammes für elektronische Schaltungen „SPICE", auch Utilities u. Quellcode.
D. Neumann, Kronenacker Straße 18, 3500 Kassel

Möglichst günstig gesucht: IMAGIC V1.1, Cyber-Paint, Cyber-Control. G.Prell, Pillenreuther Str. 105, W-8500 Nürnberg 40

Platinenlayout-Programm PCB Layout/ Adimens 2.3 bis 80,– 520 + 1040, Floppy, Monitore, Scanner, auch defekt günstig zu kaufen. M.Klein, Carl-Spitzweg-Straße 1, 7920 Meidenheim

---

That's Write Profiversion =1.3, zu kaufen gesucht. Nur Originale!
Tel. 06151/423876, ab 18 Uhr

Suche Programm zum Berechnen u. Drucken von (Solar-) Horoskopen nach Placidus. Tel. 08683/1799

Kontakte

Atari-Freaks im Raum Bremerhaven. Möchte Atari-Club gründen, wer hat Lust mitzumachen? Tel. 0471/44132

Wer hat Umbauerfahrung mit Towergehäuse für 520er und kann mir helfen.
Tel. 040/4392217, Christian

Suche Kontakt zu Unterrichtsschwestern/-pfleger, die ihren Unterricht auf dem ST erstellen. Zwecks Erfahrungsaustausch.
Tel. 040/6781648

Suche Kontakt zu ST/E Usern zwecks Programm- und Erfahrungsaustausch (DFÜ). Auch Clubs!
Postfach 15, 8303 Rottenburg

Mailbox LAO-TSE Die Box für Atari-User, Netze: Magic-Net, Zerberus, Amnet 50 MB PD-Software für alle Syst.
Tel. 09372/72321

Kleinanzeigen

Biete Hardware

Anschlußfertige **SCSI Festplatten:**
40MB/19ms oder 50MB/28ms 999,– DM
80MB/19ms 1350,– DM
160MB/19ms 2000,– DM
Tel.: 02922/82195 nach 18 Uhr

I/O-Interface ITF-32, 32 TTL-komp. Ein- u. Ausgänge f. Ind.-Steuerungen, Alarmanlage etc.
J. Sieber Raffeltersteige 15, 7100 Heilbronn, Tel. 07131/161151

Speichererweiterungen für 260/520ST 2.5 MB: DM 399,–; 4 MB: DM 749,– Einbau, 12 Monate Garantie: DM 50,– heiler engineering 0721/376622

ST-Floppy 3,5", 720KB, DM 129,–; **Mega ST-Gehäuse** DM 40,–; Mega ST-Netzteile DM 70,–; **SM124-Monitor** DM 248,–; Atari Maus DM 49,–; Mega ST-Tastatur DM 230,–; Alles neu
Tetra Computer Systeme 02225/17081

24-Nadeldrucker + Einzelblatteinzug 650,–; 30MB Festplatte Vortex HD Plus 30 650,–; Easyprommer mit Software 150,–. Tel. 0221/85192

Mega ST4+Megafile30+Megafile44+ 2*SM124+SLM804 Laser+Spectre GCR+ Init+MacPrint+MacSoftw.+ Calamus 1.09+Outlineart+200 DTPGraf.+ 40Fonts: 4Mon. alt; O. Fegers
Tel. 0211/4238 17. Preis: 9600,- VB

Atari Mega ST1 + SM124 la VB 1100,–; Signum2 280,–; Eprommer 190,–; Alles zus. mit ca. 10 kg Literat.
VB 1550,– H. Peter. Tel. 07733/6458

Mega ST2 mit Monitor + Megafile 30 + Matrixdrucker auch für Einzelblatt.
Dieter Hartig, Kasseler Straße 76, W-2800 Bremen 1

Supercharger mit 1MB + Kabel neuwertig 40,– DM. Tel. 06071/287412

Mega ST4 1900,–; s/w-Monitor SM124 150,–Tel. 09255/633, abends 09255/7480

Atari **Mega ST4**, SM124; 20MB; VB 2500,– DM; Tel. 02565/2718

Atari SM124 neu, aufgrund Sammelbestellung abzugeben. VB: 250,–. Suche Megafile 60 neu bzw. kaum gebraucht. Tel. 089/3107085

ICD Advantage Micro SCSI-Hostadapter zum Einbau in Mega ST zu verkaufen. 1 Std. alt! Festpreis: 200,-. Tel. 0211/463354

Rainbow TOS 1.4 mit ST-Computer Patches 65,-/ Schaltpläne ST/STE/Mega/ SM124/ Megafile je 25,-/ Protos V 1.1 30,– Speichererweiterung von 1 auf 3MB 350,– STE auf 2/4 MB 250/450,-.
Tel. 04351/44252

1040STF+SH205+SM124 zu verkaufen wegen Systemwechsel, Festplatte 20MB, Monitor s/w und Maus. VB 1200,– DM Tel.: 02106/41718

Mega ST, Tower, 4MB, Seagate 48MB und Syquest 44MB Fest- und Wechselplatte, 5,25" und 3,5" Laufw. 4800,– DM. Tel. 0228/211955

Fernschreiber SEL LQ2001 ESR mit elektr. Speicher 4000 Zeichen. postzugelassen u. verplombt. Nur Abholung o. Versand unfrei. VB 1500,– DM. Neupreis über 11000! Tel. 02237/2489 W. Otto

Verkaufe **Drucker NEC P6/Plus** für 950,– DM Tel: 02721/2432

1040STF, Uhr, SM124, Abdeckhaube, Easytizer-Videodigitizer DM 950,–
Auch einzeln abzugeben.
Tel. 02253/3523

Verkaufe SCSI-Festplatte im Megafile-Gehäuse, 85 MB, 24ms, 700KB Transferrate. Preis: 1200,– Tel. 0611/806913

Blitter 50,- DMs Sockel 15,- DM,
Tel. 069/5071694

Atari 520 STM mit 54MB TOS 1.2, Maus, allen Handbüchern, aber ohne Netzteil. Voll funktionsfähig, jedoch Verdacht auf Shifterdefekt. Gegen Höchstgebot zu verk. Tel. 02405/72307

Mega ST4, SH205, SM124, Maus+ Software DM 2500,–
Tel. 0202/706596, ab 18 Uhr

1040ST, PC-Geh., Lüfter, 4MB, RTS, TOS 1.4, 2. Netzteil f. Erw., SM124: 1600,–; Turbo-C 2.0 VB 250,–; ext. LW 150,–; Literatur VB, PD VB
Tel. 069/2663760, 9-17 Uhr

AT-Speed, 6 Mon. alt 400,– 2 NEC LW 720KB je 60,–; ICD-SCSI-Hostadapter 6 Mon alt 150,–; Juniorprommer 130,–; Org. Easytizer Teilesatz v. Maxon 100,–; SM124 100,–. Tel. 02406/5652

TOS 1.4 mit Patches aus ST-Computer 1+10/90 (ohne Fastload) in 6 ROMs für 60 DM bei Vorkasse, sonst + NN:
H.K. Postfach 2245, 2900 Oldenburg

Mega ST1, nur 3/4 Jahr alt, mit verbessertem Laufwerk, komplett, mit diversem Zubehör, wegen Computerwechsel nur DM 999!!! Tel. 07731/27239

Suche TV-Modulator für ST. Zahle 90,- DM. Tel. 02156/5231, ab 18 Uhr

Verkaufe: Scanner an Drucker, Epson FX inkl. Software DM 200,–; ST 3D-Grafikprogrammierung (inkl Disk); Text-Design ST Tel. 06691/23786

Drucker C.ITOH 8510k, 8 Nadeln, ink.. ausf. Unterlagen für DM 350,–; Atari 1040STF inkl. SM124 für DM 800,–; Atari-8-Bit Infos, Literatur gegen Portoerstattung abzugeben. Tel. 02451/64477, 18-20 Uhr

SM 124, gebraucht, DM 125,–
Tel. 08024/2064

2. Laufwerk Prof. DL1314 mit 720KB für 150,– DM, bitte nur schr. Anfragen an:
S. Westermann, Meuschelstr. 46, 8500 Nürnberg 10

Mega ST1, mit Monitor, Drucker, vielen Disks u. Lit. f. 1600,– DM zu verk. Sirko Braucher, Geizenberg 2, 0-5812 Waltershausen

MMU, Shifter 70,– Automat. elektron. Computerumschalter (parallel, nwtg), 5 Computer an 1 Drucker 280,– 8 Computer an 1 Drucker 350,- Orig. TOS v1.0 30,– (6 ROMs). Tel. 0631/21328

Mega ST, 4MB, SM124, PC-Speed 1.5, SH 205, 20MB, TOS 1.4, GFA-BASIC 3.5., Turbo ST, 2700,– DM; LQ850, 24 Nadeln, Tel. 05171/12499, abends

Atari 520ST+, Maus, SF314, SM124(neu) für VB 750,– DM. Tel. 089/4307487

TT, 4MB, 48MB Festplatte, nagelneu, originalverpackt 4000,–; Suche Kontakte zu ST/TT-Usern zwecks Programmtausch usw. M. Arpe 0911/534047

Verkaufe **1040STF/TOS 1.4 u. 1.0**, umschaltbar/Stereo/2.3,5" Laufwerk/SM124/ Colormonitor ITT 7080/ Maus/Umschalter/Literatur/Software. Preis VB.
Tel. 06198/7219, Andreas

Akustikkoppler **Dataphon S21D** inkl. Anschlußkabel und Terminalprogramm zu verkaufen. Tel. 089/7855365

Original Atari TOS-ROMs Version 1.0, 6 ROMs fur 30,– DM Tel. 089/7855365

2* 80MB Quantum SCSI-Festplatte, Rohlaufwerk 3,5" zu verkaufen.
Tel. 02737/3131

Megafile 60 wg. Tower-Umbau: VB DM 750,– (mit viel PD-Software) / Mega ST-Leergehäuse mit Floppy. Tel. 0228/231811

Laserprinter Atari SLM 804 gebr., wg. Wechsel, la Zustand DM 2200,–, Mwst. auswbar. C. Heitz Tel. 07336/5605,
Fax 5617

Monitor SM124 DM 150,–; PC-Speed v1.S DM 200,–; Monitorumschalter DM 30,- zu verkaufen Tel. 02738/8756, ab 18 Uhr

260ST, SM124, SF354, 1MB RAM, Maus + div. Original-Programme (Text, Grafik usw) DM 500,– oder Tausch gegen Festplatte Tel. 02151/500150

Supercharger 1MB, v 1.4, neuwertig DM 450,–, Tel. 0521/330190

Farbmonitor SC1224 VB 300,- DM oder im Tausch gegen SM124 mit Wertausgleich. Steffen Ebert, Rennebogen 101, O-3042 Magdeburg

Atari STM, 2,5MB, SM124, Uhr, RTS-Tastenkappen, 2 Doppelfloppies, PC-Speed, 1650,– DM.
Tel. 06752/8057, ab 18 Uhr

1040STF, SM124, 2*3,5"-Laufwerk, 20MB Festplatte in Lighthouse Tower abgesetzte Tastatur DM 1990,– (mit Megaclock) NEC P6 mit Bidi-Traktor DM 1099,–. Tel. 02238/14285

Star NB 24-10-Drucker mit vollautomatischem Einzelblatteinzug und Kabel, Preis VS! Tel. 0202/88093, ab 19 Uhr

Mega ST1, Festplatte SCSI 50MB, Hyperscreen, TOS 1.4, PD-Software 2050,– DM VB. Tel. 09131/54260, ab 17 Uhr

1040ST, 2,5MB, TOS 1.4, Tast-Schutz + SH205, umfangr. Softw. (PC-Ditto, Script2, Signum, NVDI, Wordplus, Protos, usw.) 1750,– VB.
Tel. 0621/376377, ab 21 Uhr

PC-Speed 1.5 neuw. 230,– DM.
Tel. 04743/6774, ab 16 Uhr

Suche Hardware

Epromprogrammiergerät für ST, Einzelblatteinzug für EPSON RX 80 FT+, Referenzhandbuch IBM Nr. 94 X 96 58
Tel.: 0681/855425

Fließkommakoprozessor 68881 bis DM 200,- gesucht. Nur vollständig Atari-kompatibel und 16 MHz! (für Mega ST).
Tel. 0441/53088, Wilfried Cordes

Suche preisgünstigen und neuwertigen **Atari Farbmonitor SC1224.**
Tel. 0201/798467

Suche Einzelblatteinzug für NEC P6.
Tel. 02281/312770, tagsüber

Handy- oder Flachbettscanner.
Tel. 04163/4488, ab 18 Uhr

Suche Monitor SM124, preiswert zu kaufen. Ihr Angebot an J. Kmitta, Kastanienstraße 3, O-8400 Riesa/Sachsen

Typenraddrucker mit Einzelblattschacht Sample Daisy Step 2000,VB 800 DM.
Tel. 0221/727092

1040STFM, 4MB, TOS 1.4, SM124, Maus, 9/89 DM 1700,–; Star NL-10 mit Einzelblatteinzug DM 530,–.
Tel. 07275/8868

PC-Speed 1.5, 1 Jahr, mit Buch, DM 290,–.
Tel. 05321/35235

Farbmonitor Highscreen 548 und SM124, inkl. Umschaltbox zusammen DM 420,–.
Tel. 07306/4666, 18-20 Uhr.

Biete Software

Artware mehr als 9000 Bilder/Graphiken auf 411 Bildschirmen aus allen Themengebieten für den ST auf 3,5" Disketten komplett für DM 69,– Tel. 0631/29707

PD-Zeichensätze für Signum,
170 Zeichensätze für 9 Nadler
oder 160 Zeichensätze
und
für 24 Nadler div. Utilities für Signum
DM 65, Tel. 0631/29707

PD-Software für Atari-Amiga-PC
Katalog kostenfrei bei:
Schreiber, 1000 Berlin 51,
Sommerstraße 37, Tel. 030/4916732

Anwender- u. Spielesoftware
Laufend n. Programme a. Lager
Riesenauswahl an Original
Soft- u. Hardware
Gratiskatalog anf. o. anrufen
W. Wunsch
Soft- u. Hardware,
07231/766595
Friedenstraße 212, 7530 Pforzheim

Preissenkung!
Softstation
2 fortl. PD von ST-Comp (1/2, usw.)
oder 1 PD ab Nr. 225 DM 4,–
bei. Komp. und sonst Serien DM 5,–
Disk. wahlw; blau, weiß, rot, grün,
gelb od. orange
Porto + NN DM 6,–
Tel.: 07195/53707

PD ab 1/80, Katalogdisk 3 DM
Hintermeier, Pf, 3551 Lahntal-Sa

Lernprogramme Mathe-, Vokabel-Grammatiktrainer: D/GB/F/E/I
Gratisinfo: I. Thurm, Pf. 1671,
W-7060 Schorndorf 07181/21709

Programmpaket, Heizungstechnik
Demodisk DM 10,– VR, J. Binder
Behrensstraße 16, 5030 Hürth

**** Querdruck2 ****
9/24 Nadeldrucker, Atari-Laser
Textdruck 90° gedreht. DM 78,–
Info: Ackermann, Kanalweg la,
W-8048 Haimhausen, 08133/1053



INHALT

TT-MANIPULATION AUF 24 BIT

Uwe Seimet

BUSFEHLER BEIM STARTEN DES TEMPUS-EDITORS AUF DEM TT? BUSFEHLER BEIM STARTEN VON CHEMPLOT? BUSFEHLER BEIM STARTEN VON ...? DEM KANN ABGEHOLFEN WERDEN. DURCH GEEIGNETE PROGRAMMIERUNG DER IM 68030 INTEGRIERTEN PMMU GEHÖREN SOLCHE PROBLEME DER VERGANGENHEIT AN.

Die meisten TT-Besitzer dürften bereits festgestellt haben, daß sich manche Programme direkt nach dem Start mit einem Busfehler (also zwei Bomben) verabschieden. Ursache für solche Fehler ist in vielen Fällen unsaubere Programmierung. War es beim ST noch möglich (aber nicht erlaubt!), die oberen 8 Bits einer Adresse für eigene Zwecke zu mißbrauchen, so schiebt der 68030-Prozessor des TT hier einen Riegel vor. Da der 68030 einen 32-Bit-Adreßbus besitzt, werden stets alle Bits einer Adresse zur Adreßberechnung eingesetzt. Beim 68000 hingegen werden die oberen 8 Bits ignoriert, da dieser Prozessor nur einen 24 Bit breiten Adreßbus besitzt.

Bereits in [1] habe ich kurz auf ein Programm hingewiesen, das sich *24BIT.PRG* nennt und von Atari Amerika entwickelt wurde. *24BIT* programmiert die PMMU (Paged Memory Management Unit), die ein Bestandteil des 68030-Prozessors ist, derart um, daß die oberen 8 Adreß-Bits ignoriert werden. So ist es möglich, auch Programme auf dem TT einzusetzen, die diese Bits für Tricks zweckentfremden.

Und ist Atari nicht willig ...

Atari Deutschland ist jedoch nicht bereit, das Programm *24BIT* der breiten Öffentlichkeit zugänglich zu machen. Diese Entscheidung wird damit begründet, daß Hilfsprogramme dieser Art in erster Linie unsauber programmierte Programme unterstützen, und daß es besser sei, so zu programmieren, daß Programme wie *24BIT* nicht benötigt werden. Unrecht hat Atari damit sicher nicht, aber andererseits bin ich der Meinung, daß es so lange sinnvoll ist, auf dieses Programm zurückzugreifen, bis alle Software-Firmen ihre Produkte TT-fest gemacht haben. Die momentane Entwicklung ist hier durchaus positiv zu bewerten.

... so brauch' ich FIX24

Wie dem auch sei, wenn *24BIT.PRG* nicht jedem TT-User zur Verfügung steht, so kann man das ja ändern, indem man sich selber ein solches Programm zusammenstrickt. Oder verwenden Sie doch gleich das hier vorgestellte Programm *FIX24.PRG*.

Was die PMMU betrifft, habe ich einige theoretische Grundlagen bereits in [1] erläutert. Insbesondere ging es dabei um die zur Adreßübersetzung erforderlichen Deskriptor-Tabellen. Auf dieser Basis stellt es keinen übermäßig großen Aufwand dar, sich mit weiteren Möglichkeiten der PMMU zu beschäftigen.

Kommt Zeit, kommt Rat

Um das 24-Bit-Problem auf dem TT zu lösen, ist es zunächst einmal wichtig, sich vor Augen zu führen , wie die Busfehler beim Zugriff auf den 32-Bit-Adreßbus eigentlich zustandekommen. Ich möchte dies am Beispiel des TEMPUS-Texteditors erläutern, da ich mich mit diesem Programm besonders intensiv beschäftigt habe, wie Sie später noch sehen bzw. lesen werden. Die Devise müßte also treffender heißen: Kommt TEMPUS, kommt Busfehler.

Für jede Textzeile existiert in TEMPUS ein Pointer, der auf den Beginn dieser Zeile zeigt. Da der MC68000 die oberen 8 Bits dieses Pointers nicht zur Adreßberechnung heranzieht, ist deren Inhalt im Normalfall ungenutzt. Der Editor versucht nun dadurch Speicherplatz zu sparen, daß er in diesem Byte noch eine weitere Information, und zwar die Länge der Zeile, unterbringt. So weit läßt sich eigentlich nichts gegen dieses Vorgehen sagen. Allerdings wird beim Ansprechen der zu einem Pointer gehörigen Textzeile nicht dafür gesorgt, daß das obere Byte für den Buszugriff ausgeblendet wird. Und genau hier liegt der Haken und somit die Ursache für den Busfehler. Das geschilderte Vorgehen bringt TEMPUS übrigens eine minimale Zeitersparnis und ist einer von vielen Gründen, warum TEMPUS zu den schnellsten Editoren für den ST gehört.

Je nach Zeilenlänge befindet sich im oberen Byte der Adresse, die der Prozessor auf den Bus ausgibt, ein Wert zwischen 0 und 255. Im Falle einer 0 passiert weiter nichts Sensationelles. Ist die Zeile jedoch genau ein Zeichen lang, führt der Speicherzugriff nicht mehr ins ST-RAM, sondern (falls vorhanden) ins TT-RAM. Dies liegt daran, daß das ST-RAM

den Adreßbereich $00xxxxxx belegt, das TT-RAM sich ab $01000000 befindet. Je nachdem, wieviel TT-RAM vorhanden ist, kann es sein, daß auch bei Zeilenlängen von 2 Bytes kein Fehler auftritt, da ja stets ein Zugriff aufs RAM (wenn auch im falschen Bereich) stattfinden kann. Bei einem Rechner mit 4 MByte TT-RAM führt ein Zugriff auf den Adreßbereich $02xxxxxx jedoch bereits zum Absturz in Form eines Busfehlers.

Außerdem bleibt festzuhalten, daß in diesem Zusammenhang auch ein Zugriff in den Adreßraum ab $01000000 nicht mehr korrekt ist. Schließlich ist hier die Adresse aufgrund der 1 im höchstwertigen Byte bereits fehlerhaft. Nur führt dieser Umstand nicht zu einem Absturz, solange der Prozessor RAM vorfindet. Lediglich alle Textzeilen der Länge 1 sind in solchen Fällen irgendwie fehlerhaft.

Die Idee

Wie man leicht erkennt, steht und fällt alles mit den 8 höchstwertigen Adreß-Bits. Könnte man den Prozessor dazu bringen, diese Bits zu ignorieren, wäre man einen Schritt weiter. Ein Zugriff auf Adressen ab $01000000 wäre dann jedoch nicht mehr möglich, da das obere Adreß-Byte ständig eine Null enthalten würde. Daraus resultiert jedoch auch der Nachteil einer solchen Notlösung: Man muß auf das schnelle TT-RAM verzichten, so daß neben dem Geschwindigkeitsverlust je nach TT-Modell nur noch 2 bzw. 4 MByte Hauptspeicher zur Verfügung stehen. Außerdem kann das in [1] vorgestellte Programm *ROM-SPEED* nicht mehr eingesetzt werden. Aber was akzeptiert man nicht alles, um den TT möglichst ST-kompatibel zu machen ...

Die Werkzeuge

Um das Programm *FIX24* zu realisieren, müssen wir uns erneut mit der PMMU des 68030 auseinandersetzen. Von den zahlreichen MMU-Registern sollte bereits der CRP (CPU Root Pointer) bekannt sein. Außerdem haben wir bereits den Aufbau der Deskriptor-Tabellen kennengelernt. Um das Gedächtnis ein wenig aufzufrischen, gibt Bild 1 einen Überblick.

Für *FIX24* ist es notwendig, sich mit weiteren Registern anzufreunden. Hierbei handelt es sich zunächst um das *CACR* (CAche Control Register). Hinzu kommen die beiden Register TT0 und TT1 (Transparent Translation-Register) sowie das TC-Register (Translation Control). Auch die uns bereits bekannte Deskriptor-Tabelle ab Adresse $700 mischt wieder mit. Wie wir ja jetzt wissen (Zitat für Insider), wird mit ihrer Hilfe der logische Adreßraum auf den physikalisch vorhandenen Speicher abgebildet.

Alle Register ziehen

Wie sind nun die benötigten Register aufgebaut? Bild 2 zeigt einen Überblick. Zunächst soll uns das *CACR* interessieren. Hier findet man die folgende Bit-Zuordnung:

WA (Write Allocate Bit): Ist dieses Bit gesetzt, wird beim Schreiben von Daten ein Cache-Eintrag erzeugt, wenn für die angesprochene Adresse ein solcher noch nicht existiert. Bei gelöschtem Bit und nicht existentem Cache-Eintrag für diese Adresse wird der Cache nicht verändert.

DBE (Data Burst Enable Bit): Dieses Bit kontrolliert, ob der Cache im burst mode des 68030 gefüllt werden soll oder nicht. Wir erinnern uns: Der *burst mode* wird vom TT-RAM unterstützt und sorgt dafür, daß der Prozessor-Cache besonders schnell gefüllt wird.

CD (Clear Data Cache Bit): Sobald das CD-Bit gesetzt wird, werden alle Einträge im Daten-Cache gelöscht.

CED (Clear Data Cache Entry Bit): Ein einzelner Cache-Eintrag kann durch das Setzen dieses Bits gelöscht werden. Der Index dieses Eintrags muß vorher in einem speziellen Register, dem *CAAR* (CAche Address Register), vermerkt werden.

FD (Freeze Data Cache Bit): Soll der aktuelle Inhalt des Daten-Cache eingefroren werden, muß das FD-Bit gesetzt werden. Sofern Daten, die sich im Cache befinden, durch Schreibzugriffe verändert werden, wird der Cache natürlich weiterhin aktualisiert.

ED (Enable Data Cache Bit): Durch Setzen dieses Bits kann der Daten-Cache aktiviert werden, Löschen des ED-Bits sorgt für das Abschalten des Daten-Caches.

IBE, CI, CEI, FI, EI: Diese Bits korrespondieren mit den bereits besprochenen Bits. Alle Funktionen beziehen sich hier lediglich auf den Befehls- und nicht auf den Daten-Cache.

Die nicht verwendeten Bits des *CACR* sind übrigens reserviert und dürfen nicht gesetzt werden.

Totale Kontrolle

Kommen wir zum Translation Control-Register *TC*. Auch hier eine Zusammenfassung der Bedeutung der belegten Bits:

E (Enable Bit): Ist dieses Bit gelöscht, findet keine Adreßübersetzung statt. Sobald das E-Bit gesetzt wird, beginnt die MMU mit ihrer Arbeit.

SRE (Supervisor Root Enable Bit): Sollen Zugriffe aus dem Supervisor-Modus anders übersetzt werden, als es für Zugriffe aus dem User-Modus der Fall ist, muß dieses Bit gesetzt werden. Der *SRP* (Supervisor Root Pointer) enthält für diesen Fall

LU	Index-Limit	
0 0 0 0 0 0 0 0 0 0 0 0 0 0		DT
Tabellen-Adresse, Bits 31-16		
Tabellen-Adresse, Bits 15-4		Unbenutzt

CPU Root Pointer Register

Tabellen-Adresse, Bits 31-16				
Tabellen-Adresse, Bits 15-4	U	WP	1	0

Tabellen-Deskriptor, kurzes Format

Seiten-Adresse, Bits 31-16								
Bits 15-4	0	CI	0	M	U	WP	0	1

Seiten-Deskriptor, kurzes Format

Bild 1: CRP und Deskriptor-Tabellen

E	0	0	0	0	0	S	F	PS	IS
TIA				TIB				TIC	TID

Translation Control Register

Adreßbasis								Adreßmaske			
E	0	0	0	0	CI	RW	M	0	FCB	0	FCM

Transparent Translation Register

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	WA	DBE	CD	CED	FD	ED	0	0	0	IBE	CI	CEI	FI	EI

Bild 2: CACR, TC, TT0/TT1

die Adresse einer Übersetzungstabelle. Enthält das *SRE*-Bit eine Null, werden User- und Supervisor-Zugriffe in der Regel gleich behandelt. Dies hängt jedoch auch vom *FCL*-Bit ab.
FCL (Function Code Lookup Bit): Ist dieses Bit gesetzt, findet die Adreßübersetzung in Abhängigkeit vom Zustand der Function-Code-Bits (FC2-FC0) statt. Hierdurch wird es möglich, Zugriffe auf Daten- und Programmbereiche im User- und Supervisor-Modus unterschiedlich zu übersetzen.
PS (Page Size): Diese vier Bits bestimmen die Größe einer Speicherseite gemäß folgender Zuordnung:

1000	256	Bytes	
1001	512	Bytes	
1010	1024	Bytes	
1011	2048	Bytes	
1100	4096	Bytes	
1101	8192	Bytes	
1110	16384	Bytes	
1111	32768	Bytes	(normal für TT)

Je nach Wert des PS-Feldes wird also eine gewisse Zahl der niederwertigen Adreß-Bits nicht für die Übersetzung herangezogen. Eine Seitengröße von 32 kByte fordert seitens der MMU den geringsten Verwaltungsaufwand.
IS (Initial Shift Count): Ähnlich wie das PS-Feld definiert IS Bits, die nicht zur Adreßübersetzung verwendet werden. Hier handelt es sich jedoch nicht um die niederwertigen, sondern um die höchstwertigen Adreß-Bits.
TIA, TIB, TIC, TID: Diese Felder legen fest, wieviele Bits zur Indizierung der einzelnen Ebenen der Deskriptort-Tabelle herangezogen werden.

Last but not least

Abschließend widmen wir uns noch den Transparent-Translation Registern TT0 und TT1. (Diese beiden Register existieren nur beim 68030, nicht jedoch bei der externen PMMU 68851.) Mit ihrer Hilfe ist es möglich, Speicherfenster zu definieren, die nicht von der Adreßübersetzung betroffen sind. Die Adreßbasis definiert dabei den Beginn einer solchen Adresse, der auf einer 16 MByte-Grenze liegen muß. Über die Adreßmaske kann die Größe dieses Bereichs eingestellt werden, und zwar in Schritten von 16 MByte. Die restlichen Bits von TT0 bzw. TT1 sind wie folgt belegt:

E (Enable Bit): Mit diesem Bit werden TT0/TT1 aktiviert.
CI (Cache Inhibit Bit): Soll der Cache innerhalb der transparenten Speicherfenster nicht verwendet werden, muß dieses Bit gesetzt werden.
R/W (Read/Write Bit): In Verbindung mit dem *RWM*-Bit kann die MMU so eingestellt werden, daß nur Lese- oder Schreibzyklen übersetzt werden. Eine 1 in diesem Bit besagt, daß nur Lesezyklen von der Übersetzung betroffen sind. Ist das *R/W*-Bit gelöscht, werden nur Schreibzyklen übersetzt.
RWM (Tja, über die Bedeutung des *M* in diesem Kürzel rätsele ich immer noch ...): Natürlich muß es möglich sein, nicht wahlweise entweder Lese- oder Schreibzyklen übersetzen zu lassen, sondern alle Zugriffe auf den Speicher. Dies ist der Fall, wenn das *RWM*-Bit gesetzt ist.
FCB (Function Code Base): Diese Bits geben an, welchen Inhalt die Function Code-Bits haben müssen, damit eine Adreßübersetzung im transparenten Speicherbereich erfolgt.
FCM (Function Code Mask): Falls gewisse Function Code-Bits für die Übersetzung nicht relevant sein sollen, können diese anhand der *FCM*-Maske ausmaskiert werden.

So, damit hätten wir die für uns interessanten Register abgehandelt. Aber keine Sorge: Es gibt noch weitere, wobei die restlichen nicht von Bedeutung sind. Was die angesprochenen Register betrifft, so ist es in diesem Rahmen nicht möglich, deren Bedeutung erschöpfend zu behandeln. Für ausführliche Erklärungen sollte man auf [2], [3] oder [4] zurückgreifen.

Nun zur Funktionsweise von *FIX24*. Bevor der Speicher vom Programm durcheinandergewürfelt wird, muß zunächst der Cache gelöscht und desaktiviert werden. (Sobald der Desktop aufgebaut wird, wird der Cache wieder eingeschaltet, falls dies in der Desktop-Info vermerkt ist.) Anschließend wird die Adreßübersetzung zunächst gestoppt. Hierzu genügt es, das Enable-Bit im *TC*-Register zu löschen. Da der 68030 es nicht unterstützt, MMU-Register per *CLR*-Befehl zu löschen (die externe PMMU 68851 bietet diese Möglichkeit), wird der benötigte Wert aus dem Speicher geholt. Da alle Adreßbereiche in die ersten 16 MByte abgebildet werden sollen, müssen noch beide Transparent Translation-Register gelöscht werden. Auch hier werden die benötigten Daten aus dem Speicher geholt.

Nachdem die Adreßübersetzung nicht mehr aktiv ist, können die neuen Übersetzungstabellen aufgebaut werden. Deren Struktur ist recht einfach, da alle logischen Adressen auf den gleichen physikalischen Bereich abgebildet werden.

Um nun die Adreßübersetzung mit den neuen Tabellen zu starten, wird das Translation Control-Register entsprechend geladen. *PS* und *IS* behalten die im Normalbetrieb des TT gültigen Werte bei, das Enable-Bit muß auf 1 gesetzt sein, damit eine Adreßübersetzung stattfindet.

Sind alle Aktionen abgeschlossen, wird *FIX24* über einen Aufruf der Funktion *PTERMRES* verlassen, bei dem als Zahl der resident zu haltenden Bytes eine Null übergeben wird. Dieses seltsame Vorgehen hat jedoch seine Gründe. Zwar hat das eigentliche Programm seine Arbeit getan und wird nicht mehr benötigt, aber der mittels *MXALLOC* angeforderte Speicher (also das gesamte TT-RAM) darf nicht wieder freigegeben werden. Dies wäre jedoch bei einem Programmende über *PTERM0* oder *TERM* der Fall.

Da nicht jeder Leser im Besitz eines Assemblers für den 68030 sein dürfte, ist neben dem Assembler-Quelltext von *FIX24* auch ein kleines Programm in GFA-BASIC abgedruckt, das die entsprechende Programmdatei direkt erzeugt. Apropos Assembler: Beim Arbeiten mit dem MAS von Borland (im Lieferumfang zu TURBO C 2.0 Professional enthalten) mußte ich feststellen, daß der MAS fehlerhafte Opcodes für Befehle erzeugt, die die beiden Transparent Translation-Register ansprechen. Besitzer des MAS sollten aus diesem Grund gar nicht erst versuchen, *FIX24.S* mit diesem Assembler zu assemblieren, sondern gleich auf das BASIC-Programm zurückgreifen.

Was *FIX24* betrifft, überprüft das Programm vor den MMU-Aktionen natürlich, ob es überhaupt auf einem TT gestartet wurde. Ist das TT-RAM nicht leer, wird keine Aktion vorgenommen, da dies zum Absturz führen könnte. Schließlich würde ein residentes Programm, welches sich im TT-RAM befindet, auf Nimmerwiedersehen verschwinden. *FIX24* muß deshalb aus dem AUTO-Ordner heraus gestartet werden, da nämlich spätestens der TT-Desktop Gebrauch vom TT-RAM macht.

Übrigens: Der Programm-Header von *FIX24* darf auf keinen Fall so manipuliert werden, daß das Programm ins TT-RAM geladen wird. Andernfalls stellt *FIX24* natürlich fest, daß dieser RAM-Bereich nicht leer ist und kann sich somit nicht installieren.

Problemfall TEMPUS

Nachdem Programme, die die oberen 8 Adreß-Bits mißbrauchen, nun auf dem TT einsatzfähig sind, werden einige Leser vielleicht versuchen, den TEMPUS Texteditor in der Version 2.10 wieder hervorzukramen. Schließlich sollte es möglich sein, TEMPUS mit Hilfe von *FIX24* auf dem TT ohne Absturz zum Laufen zu bringen. In der Tat geschieht nichts Ungewöhnliches, wenn

KLASSE statt MASSE

Dieter und Jürgen Geiß

Vom Anfänger zum GEM-Profi

Perfekte Programmierung auf Atari ST und IBM-PC

2., überarb. Auflage 1991, ca. 550 S., geb., DM 98,– incl. Diskette ISBN 3-7785-2049-0

Entwickler müssen mit diesem Titel nur noch minimalen Aufwand betreiben, um perfekte Software unter GEM zu erstellen. Ermöglicht z.B. Pop-Up-Menüs oder Menüzeilen in Fenstern, Accessories mit eigenem Desktop und Menüzeile. Mit Richtlinien zur GEM-Programmierung, empfohlen von Atari und Digital Research.

Christiane und Jürgen Kehrel

Omikron-BASIC

Befehle, Bibliotheken, Utilities

1989, 590 S., geb., DM 58,– ISBN 3-7785-1662-0

Jeder BASIC-Befehl (auch die undokumentierten) wird präzise mit einem Syntaxdiagramm und einem Beispiel erklärt. Es folgen BIOS-, XBIOS- und GEMDOS-Aufrufe mit einer Funktions- und Parameterbeschreibung. Auch die komplette GEM.LIB (AES und VDI) wird auf 150 Seiten so beschrieben, wie sie wirklich unter OMIKRON.BASIC ansprechbar ist. Erläuterungen der wichtigsten Bibliotheken und der verfügbaren Hilfsprogramme runden das Buch ab.

Ralf J. Schläfer

OMLib Professional

1991, Handbuch + Diskette, ca. DM 129,– ISBN 3-7785-2050-4

OMIKRON-Library rund um einen neuen Form-Do-Manager mit über 100 neuen Befehlen zur einfachen GEM-Programmierung. Mit speziellem Resource-Construction Set.

BESTELLCOUPON

einsenden an: Hüthig Buch Verlag GmbH
Im Weiher 10, 6900 Heidelberg

Titel

Name, Vorname

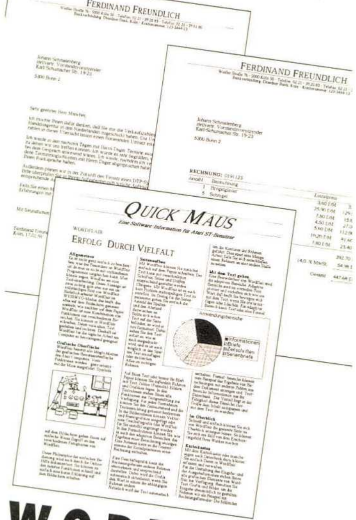
Straße, Nr.

PLZ, Ort

Datum, Unterschrift



WUNDER?



WORDFLAIR

Wordflair ist mehr als eine Textverarbeitung! Text, Grafik, Kartei- & Rechenfunktionen praxisgerecht in einem Programm. DM 249,-*

Test: STC10/90, PDJ12/90, STM 11/90, TOS 10/90.

NEODESK

„Vollkommener" Desktop. Vielfalt, die das tägliche Leben noch leichter macht. Neodesk 3 DM 98,-* CLImax Befehlsinterpreter DM 69,-*

Test: STC 11/90, PDJ 12/90

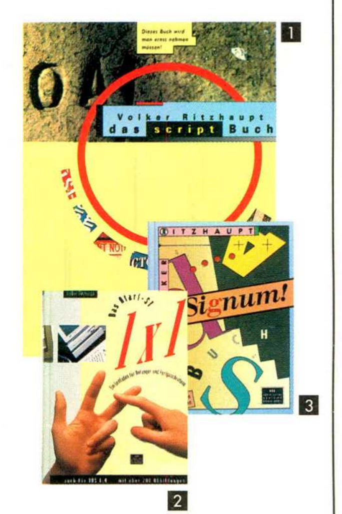
*unverbindliche Preisempfehlung

COMPUTERWARE

Im gut sortierten Fachhandel oder bei:
Computerware • G. Sender • Weißer Str. 76 • 5000 Köln 50
Telefon: 02 21 - 39 25 83 • Telefax: 02 21 - 39 61 86
Schweiz: Data Trade AG Zürich, Telefon 0 56 - 82 18 80
Österreich: Reinhart Temmel GmbH, Telefon 06 62 - 71 81 64

mmad

buch müssen bescheid wissen*



*sagt Thomas C. Breuer.
Diese Bücher wissen Bescheid:

1. **Das Script Buch**, 332 Seiten, gebunden, 49,-- DM **neu**
2. **Das Atari 1 x 1**, 320 Seiten, gebunden, 49,-- DM
3. **Das Signum! Buch**, 432 Seiten, gebunden, 59,-- DM

und natürlich, im Großformat:
351 Signum!-Zeichensätze, 232 Seiten, Pb., 29,-- DM
547 neue Signum!-Zeichensätze, 424 Seiten, Pb., 39,-- DM

Man kriegt sie bei uns.



Application Systems Heidelberg
Englerstraße 3, Postfach 10 26 46
D-6900 Heidelberg 1
Telefon (0 62 21) 30 00 02
Fax (0 62 21) 30 03 89

man TEMPUS nach der Installation von *FIX24* startet. Beim Arbeiten treten zunächst keine Probleme auf. Verläßt man TEMPUS jedoch, gibt es einen dieser besonders netten TT-Abstürze, bei denen der Bildschirm im Nirwana verschwindet und gar nichts mehr geht.

Woran das nun wieder liegt? Nun, TEMPUS besitzt einen integrierten Bildschirmschoner, der sich eines Verfahrens bedient, das man seinem Monitor zuliebe nicht verwenden sollte: Zum Abschalten des Bildschirms wird auf externe Bildsynchronisation geschaltet. Beim ST sorgt dies dafür, daß das Bild verschwindet. Beim TT dagegen kann bei Manipulationen des entsprechenden Bits der Adresse $FF820A ein Absturz die Folge sein. Selbst wenn der Bildschirmschoner von TEMPUS abgeschaltet ist, greift das Programm beim Verlassen auf das bewußte Bit zu. Das Ergebnis ist uns ja nun bekannt.

Ein kleiner, wenn auch nicht ganz sauberer Patch schafft hier Abhilfe. Zugriffe auf die Adresse $FF820A kommen innerhalb von TEMPUS nicht übermäßig häufig vor. Mit geringem Aufwand ist es möglich, dem Editor einfach eine andere, ungenutzte Adresse unterzujubeln. Alles, was man dazu braucht, ist ein Diskmonitor, mit dem man alle Byte-Folgen $FF820A innerhalb der TEMPUS-Programmdatei aufsucht und durch $001000 ersetzt. ($1000 stellt eine ungefährliche Dummy-Adresse dar, aber gerade das macht diesen Patch unsauber.) Nach dieser kosmetischen Operation kann TEMPUS auf dem TT in der mittleren und hohen ST-Auflösung benutzt werden. Ist doch eigentlich ganz einfach, oder?

Übrigens soll es demnächst eine TEMPUS-Version geben, für die solche Kunstgriffe nicht mehr nötig sind. Über diese Version wird sich sicherlich auch die Firma Atari freuen, zeigt es doch, daß trotz *24-BIT.PRG* und *FIX24.PRG* Programme konsequent TT-tauglich gemacht werden.



Literatur:

[1] „Speed without the price - TT-Tuning", ST-Computer 3/91

[2] „MC68030 32-Bit Microprocessor User's Manual", Motorola Inc.

[3] „MC68851 Paged Memory Management Unit User's Manual", Motorola Inc.

[4] Steve Williams, „68030 Assembly Language Reference", Addison-Wesley Publishing Company Inc.


```
  1:  ***************************************
  2:  *                                     *
  3:  * FIX24.PRG                           *
  4:  *                                     *
  5:  * beschränkt Adressierung auf 24 Bit *
  6:  *                                     *
  7:  * by Uwe Seimet  (c) MAXON Computer  *
  8:  *                                     *
  9:  ***************************************
 10:
 11:
 12:  GEMDOS   = 1
 13:  CCONWS   = 9
 14:  SUPER    = 32
 15:  PTERMRES = 49
 16:  MXALLOC  = 68
 17:  TERM     = 76
 18:
 19:
 20:  _p_cookies = $5a0                     ;Pointer auf
 21:                                        ;cookie-jar
 22:
 23:
 24:  magic    = $1357bd13
 25:
 26:
 27:           text
 28:
 29:           clr.l -(sp)
 30:           move #SUPER,-(sp)
 31:           trap #GEMDOS
 32:           addq.l #6,sp
 33:           move.l d0,d7
 34:           moveq #-1,d6                 ;Fehlerflag
 35:                                        ;setzen
 36:           lea sterr(pc),a6
 37:           move.l _p_cookies,d0
 38:           beq quit                     ;kein
 39:                                        ;cookie jar-
 40:           move.l d0,a0
 41:  cloop:   movem.l (a0)+,d0-d1
 42:           tst.l d0
 43:           beq quit                     ;Ende des
 44:                                        ;cookie jar-
 45:           cmp.l #"_MCH",d0             ;Computertyp?
 46:           bne cloop                    ;nein-
 47:           swap d1
 48:           subq.l #2,d1                 ;TT?
 49:           bne quit                     ;nein-
 50:           cmp.l #magic,$5a8            ;TT-RAM
 51:                                        ;vorhanden?
 52:           bne.b noram                  ;nein-
 53:           move.l $5a4,d0               ;Ende des
 54:                                        ;TT-RAM
 55:           cmp.l #$1000000,d0
 56:           ble.b noram                  ;kein
 57:                                        ;TT-RAM-
 58:           lea ramerr(pc),a6
 59:           sub.l #$1000000,d0           ;ergibt Größe
 60:                                        ;des TT-RAM
 61:           move #1,-(sp)                ;TT-RAM
 62:           move.l d0,-(sp)              ;belegen
 63:           move #MXALLOC,-(sp)
 64:           trap #GEMDOS
 65:           addq.l #8,sp
 66:           tst.l d0
 67:           beq.b quit                   ;TT-RAM ist
 68:                                        ;nicht leer-
 69:  noram:   movec.l cacr,d0              ;Cache
 70:           or #$0808,d0                 ;löschen
 71:           and #$fefe,d0                ;und
 72:           movec.l d0,cacr              ;abschalten
 73:           pmove dummy,tc               ;Übersetzung
 74:                                        ;anhalten
 75:           pmove dummy,tt0
 76:           pmove dummy,tt1
 77:           lea $0700,a0
 78:           moveq #15,d0
 79:  fill1:   move.l #$00000742,(a0)+      ;Ebene 0
 80:           dbra d0,fill1
 81:           moveq #15,d0
 82:  fill2:   move.l #$000007c2,(a0)+      ;Ebene 1
 83:           dbra d0,fill2
 84:           moveq #1,d1
 85:           moveq #14,d0
 86:  fill3:   move.l d1,(a0)+              ;Ebene 2
 87:           add.l #$00100000,d1
 88:           dbra d0,fill3
 89:           move.l #$00f00041,(a0)       ;für ROM-
 90:                                        ;Übersetzung
 91:           pmove trans,tc               ;Übersetzung
 92:                                        ;neu starten
 93:           lea message(pc),a6
 94:           clr d6                       ;Fehlerflag
 95:                                        ;zurücksetzen
 96:  quit:    move.l d7,-(sp)
 97:           move #SUPER,-(sp)
 98:           trap #GEMDOS
 99:           addq.l #6,sp
100:           pea (a6)
101:           move #CCONWS,-(sp)
102:           trap #GEMDOS
103:           addq.l #6,sp
104:           move d6,-(sp)
105:           bmi.b error
106:           clr.l -(sp)
107:           move #PTERMRES,-(sp)
108:           trap #GEMDOS
109:
110:  error:
111:           move #TERM,-(sp)
112:           trap #GEMDOS
113:
114:
115:  trans:   dc.l $80f04445     ;für TC-Register
116:
```

→

```
117:  sterr:   dc.b $0d,$0a,"FIX24 läuft nur "
118:           dc.b "auf dem TT!",$0d,$0a,$00
119:  message: dc.b $0d,$0a,"FIX24 V1.0 "
120:           dc.b "installiert",$0d,$0a
121:           dc.b "(c) 1991 by MAXON",$0d,$0a,$00
122:  ramerr:  dc.b $0d,$0a,"Das TT-RAM ist "
123:           dc.b "nicht leer!",$0d,$0a,$00
124:
125:
126:           bss
127:
128:  dummy:   ds.l 1              ;für TC-Register
```

Listing 1: FIX24.S

```
 1:  OPEN "O",#1,"FIX24.PRG"
 2:  FOR i=1 TO &H18E
 3:    READ byte
 4:    PRINT #1,CHR$(byte);
 5:  NEXT i
 6:  CLOSE #1
 7:  DATA &60,&1A,&00,&00,&01,&6A,&00,&00,&00,&00,&00,
      &00,&20,&04,&00,&00
 8:  DATA &00,&00,&00,&00,&00,&00,&00,&00,&00,&01,&00,
      &00,&42,&A7,&3F,&3C
 9:  DATA &00,&20,&4E,&41,&5C,&8F,&2E,&00,&7C,&FF,&4D,
      &FA,&00,&EA,&20,&38
10:  DATA &05,&A0,&67,&00,&00,&B8,&20,&40,&4C,&D8,&00,
      &03,&4A,&80,&67,&00
11:  DATA &00,&AC,&B0,&BC,&5F,&4D,&43,&48,&66,&EE,&48,
      &41,&55,&81,&66,&00
12:  DATA &00,&9C,&0C,&B8,&13,&57,&BD,&13,&05,&A8,&66,
      &28,&20,&38,&05,&A4
13:  DATA &B0,&BC,&01,&00,&00,&00,&6F,&1C,&4D,&FA,&00,
      &FD,&90,&BC,&01,&00
14:  DATA &00,&00,&3F,&3C,&00,&01,&2F,&00,&3F,&3C,&00,
      &44,&4E,&41,&50,&8F
15:  DATA &4A,&80,&67,&68,&4E,&7A,&00,&02,&80,&7C,&08,
      &08,&C0,&7C,&FE,&FE
16:  DATA &4E,&7B,&00,&02,&F0,&39,&40,&00,&00,&00,&01,
      &6A,&F0,&39,&08,&00
17:  DATA &00,&00,&01,&6A,&F0,&39,&0C,&00,&00,&00,&01,
      &6A,&41,&F8,&07,&00
18:  DATA &70,&0F,&20,&FC,&00,&00,&07,&42,&51,&C8,&FF,
      &F8,&70,&0F,&20,&FC
19:  DATA &00,&00,&07,&C2,&51,&C8,&FF,&F8,&72,&01,&70,
      &0E,&20,&C1,&D2,&BC
20:  DATA &00,&10,&00,&00,&51,&C8,&FF,&F6,&20,&BC,&00,
      &F0,&00,&41,&F0,&39
21:  DATA &40,&00,&00,&00,&00,&F6,&4D,&FA,&00,&4E,&42,
      &46,&2F,&07,&3F,&3C
22:  DATA &00,&20,&4E,&41,&5C,&8F,&48,&56,&3F,&3C,&00,
      &09,&4E,&41,&5C,&8F
23:  DATA &3F,&06,&6B,&08,&42,&A7,&3F,&3C,&00,&31,&4E,
      &41,&3F,&3C,&00,&4C
24:  DATA &4E,&41,&80,&F0,&44,&45,&0D,&0A,&46,&49,&58,
      &32,&34,&20,&6C,&84
25:  DATA &75,&66,&74,&20,&6E,&75,&72,&20,&61,&75,&66,
      &20,&64,&65,&6D,&20
26:  DATA &54,&54,&21,&0D,&0A,&00,&0D,&0A,&46,&49,&58,
      &32,&34,&20,&56,&31
27:  DATA &2E,&30,&20,&69,&6E,&73,&74,&61,&6C,&6C,&69,
      &65,&72,&74,&0D,&0A
28:  DATA &BD,&20,&31,&39,&39,&31,&20,&62,&79,&20,&55,
      &77,&65,&20,&53,&65
29:  DATA &69,&6D,&65,&74,&0D,&0A,&00,&0D,&0A,&44,&61,
      &73,&20,&54,&54,&2D
30:  DATA &52,&41,&4D,&20,&69,&73,&74,&20,&6E,&69,&63,
      &68,&74,&20,&6C,&65
31:  DATA &65,&72,&21,&0D,&0A,&00,&00,&00,&00,&7C,&08,
      &08,&3A,&00
```

Listing 2: FIX24.LST



DIE EINZIGARTIGE ALTERNATIVE ZUR WECHSELPLATTE
220 V AC
ON
OFF
DEVIVE I/O
SCSI I
SCSI II
DMA IN
DMA OUT
SCSI IN
SCSI OUT
* ICD ADVANTAGE- ADAPTER * ANSCHLUSS
* WIE HARDDISC * GEHT MIT TOS 1.0 BIS 1.6
* 100% ATARI- UND AHDI- 3.XX- KOMPATIBEL
* 256 MB/PARTITION MAX (THEOR.)
* 14 PARTITIONEN MAXIMAL * BOOTPART. FREI WAEHLBAR * KOMPATIBEL ZU: VORTEX, PC-, AT-SPEED,SUPERCHARGER, PC-DITTO,SPECTRE, ALADIN U.A. * HARDWARE-SCHREIBSCHUTZ
* SOFORT BOOT-FAEHIG * SPURENCACHING 16 KB * EINZIGARTIGE FEHLERKORREKTUR UND SICHERHEIT * SEHR HOHE LEBENSDAUER VON MEDIUM UND GERAET * EINFACHSTE BEDIENUNG * DEUTSCHES HANDBUCH
1 JAHR GARANTIE * RUECKGABE - RECHT ****
MEDIUM 5 1/4 ZOLL
DATEN | BESCHREIBUNG
250-300 KB/S | UEBERTRAGUNGSRATE DURCHSCHNITTLICH
20,05 / 24 MB | SPEICHERKAPAZITAET FPRMATIERT/UNFORMATIERT PRO DISC
55-65 MS | MITTLERE ZUGRIFFSZEIT
JA | MEDIA-DISC-CHANGE PER ESCAPE-TASTE
JA - 0 UND 1 | DMA-ADRESSE VON HINTEN EINSTELLBAR
JA | DMA-BUS GEPUFFERT IN/OUT
OPTION | SCSI-BUS HERAUSGEFUEHRT
GEHAEUSE | ATARI-MEGA-DESIGN MIT ABGERUNDETEN KANTEN - GRAU
NETZTEIL | VDE-SCHALTNETZTEIL, REICHT AUS FUER ZWEI SYSTEME
INTERFACE | ICD-HOSTADAPTER, KOMPL. SCSI-BEFEHLSSATZ NACH ANSI
ECHTZEITUHR | JE NACH WUNSCH OHNE AUFPREIS DMA-ADRESSE 6
SOFTWARE | KOMPL. SOFTWARE, GENAUSO WIE FUER FESTPLATTEN ANBEI
EMPFINDLICHK. | WESENTL. UNEMPFINDLICHER GEGEN SCHOCK,STAUB ALS HARDDISC
LEBENSDAUER | 10-30 MAL HOEHER ALS BEI SEHR GUTEN FESTPLATTEN
LUEFTER | KEIN LUEFTER VORH, EXTREM LEISES LAUFWERK
ERROR CORREKT | AUTOM. FEHLERKORREKTUR+ERKENNUNG,SPUREN-CHACHING
GRUNDAUSTATTUNG : SCSI-SCHALTER
AUSBAUSTUFE 1 SCSI-BUS extern
AUSBAUSTUFE 2 SCSI-BUS in-out
AUSBAUSTUFE 1 : +100 DM, 2: +200 DM
SIEHE ST_COMP.11-90
Seite 60 TEST !
DM : 1898.-
MEDIUM: 79 DM
MIT VERBATIM-LAUFWERK
AT - SPEED : 449 DM V: 2.22
ATARI TT (32 MHz)
mit VGA-MULTISYNC,ab 4MB HAUPTSPEICHER:
OHNE MONITOR UND OHNE FESTPLATTE
SOFORT LIEFERBAR!
DAFÜR EINBAU NACH WUNSCH:
FEST- ODER WECHSELPLATTE!
ATARI-PAKETE: SM 124 + MEGAFILE 30 + WORD PERFECT:
MEGA-I- PAKET: 1998 DM
MEGA-2- PAKET: 2198 DM
MEGA-1 +SM124 solo :1.398 DM
MEGA-2 +SM124 solo :1.598 DM
SM 194 Bigscreen: 3698DM
ST-3 720 KB DISKETTENLAUFWERK (TEAK) 199 DM
ST-5 360,720,1.2 MB 5.25"Laufw. (TEAK) 299 DM
Komplett incl Netzteil,Gehause, Kabel ect
FISCHER COMPUTER GOETHE-7 6101 FR.-CRUMBACH 06164-4601 FAX: 3748
SCSI FESTPLATTEN
NOCH MODERNER UND ZUVERLÄSSIGER
SUPER AUSSTATTUNG
ALLE GERATE WERDEN KOMPLETT ANSCHLUSSFERTIG AUSGELIEFERT !!
MEGA-ST-STAHLBL GEH GRAU, NETZKABEL 2m, DMA - KABEL, HANDBUCH, KURZANLEITUNG, DISKETTEN
ICD-ADAPTER ADVANTAGE ICD-TREIBER-UND APPLICATION-SOFTWARE
100 % ATARI-ST --- ATARI-TT KOMPATIBEL AHDI 3.xx
MS-DOS-KOMPATIBEL PC-SPEED, AT-SPEED, SUPERCHARGER, PC-DITTO,IBM-KOMP
SPECTRE, ALADIN, OS 9, RTOS, MINIX
SCSI-BUS EXTERN UND DMA_ADR-SCHALTER extra!
VORTEX, LASERDRUCKER, SCANNER--
keine Probleme !!!
NETZWERKFÄHIG !!!
LÄUFT UNTER TOS 1.0 bis 2.0
mit allen Rechnern.
SCSI-ADRESSE :
DMA: 0-7
DMA GEPUFFERT IN - OUT
AUTOPARK HARDWAREMÄSSIG
64 kB CACHE_SPEICHER 64 kB SOFTWAREMASSIG, 32 kB HARDWAREMASS
AUTOBOOTFÄHIG VON ALLEN PARTITIONEN, BOOT FREI WAHLBAR - ACCESSORYSTEUERUNG (GEM)
14 PARTITIONEN UNTER TOS, PARTITIONEN EINZELN PARTITIONIERBAR (TOS,DOS..)
KONSTANTGEREGELTER LÜFTER EINZIGARTIGE RETRY - VERIFY - FUNKTION
SUPERLEISE -- NEUESTE LAUFWERKE
EINBAU EINES ZWEITEN CHASSIS - JA !
ALLE PLATTEN HABEN INTERLEAVE 1-1
MHD 50 48 MB (SEAGATE ST 157N)28 ms, 620kB-s 3.5 zoll 998 DM
MHD 80 85 MB (SEAGATE ST 296N)28 ms, 550kB-s 5.25 zoll 1.098 DM
MHD 81 84 MB (S.-IMPRIMIS 1096N)24 ms, 770kB-s 3.5 zoll 1.248 DM
MHD 140 140MB ("-") NUR 15 ms, 1020KB-s,32kB CACHE 3.5 zoll 1.998 DM
MHD 170 170MB ("-") NUR 15 ms, 1200KB-s,64kB CACHE 3.5 zoll 2.398 DM
MHD 210 210MB ("-") NUR 15 ms, 1250KB-s,64kB CACHE 3.5 zoll 2.545 DM
SCSI-BUS: 50 DM
DMA_SCHALTER:
AUFPREIS: 25 DM
neu QUANTUM 1 zoll
17 ms, 1440 kB-s, 64+64 kB CACHE, prog bar
besser und schneller als die alten Quantum-Platten
MHD 52 PRO 52 MB 1.498 DM
MHD 105PRO 105MB 1.898 DM
DIE PRO-SERIE WIRD MIT Backup-Programm plus Optimizer ausgeliefert
Alle Platten sofort lieferbar
2 Jahre Garantie !!
Michael Fischer • Computersysteme
Goethestr. 7 • 6101 Fränkisch-Crumbach • Tel. 0 61 64 / 46 01

KOMPAKTER TEXTEDITOR ALS MODUL

Rolf Stelljes

KENNEN SIE DAS? SIE MÖCHTEN IN IHREM PROGRAMM MEHRZEILIGE TEXTEINGABEN ERMÖGLICHEN, WOBEI Z.B. DIE ZEILENLÄNGE AUFGRUND DER BILDSCHIRMAUFTEILUNG BEGRENZT IST ODER SICH SOGAR VON FALL ZU FALL ÄNDERT.

Kein Problem ... mit *FORM INPUT* müßte es gehen: Zeilenlänge wird angegeben, BACKSPACE und DELETE geht, Cursor hin und her geht auch. Aber was ist mit Cursor rauf und runter oder den Funktions- oder Sondertasten? Dann ist da noch die Sache mit dem Bildschirm-Scrolling und der Speicherverwaltung; außerdem soll das Hauptprogramm nicht unnötig aufgebläht werden, und die Textzeilen müssen leicht zugänglich sein. Doch ein Problem ? Nicht mit *ED_IT*!

Das nachstehende Modul erfüllt die gestellten Anforderungen auf einfache und durchschaubare Weise: Tastendruck abwarten - Taste auswerten - falls ASCII-Zeichen, dieses an Zeile anhängen - falls Steuertaste, Aktionen ausführen - fertig! Cursor- und Bildschirmssteuerung werden mit VT52-Sequenzen erledigt, so daß das ganze „Grafikverwaltungs- und Cursor-wo-bist- Du-Geraffel" entfällt. Diese Sequenzen werden normalerweise für die Ansteuerung von Bildschirmterminals des Typs 52 verwendet. Solche Terminals werden in Großrechneranlagen verwendet, wo mehrere Arbeitsplätze mit einem Zentralcomputer verbunden sind, und bestehen eigentlich nur aus Bildschirm, Tastatur und Zwischenspeicher. In den empfangenen Texten, die auf den Bildschirm sollen, sind bestimmte Steuerzeichen versteckt, mit denen z.B. Cursor-Position und Schriftfarbe verändert werden oder Teile des Bildspeichers gelöscht werden können. Sie bestehen aus einer Folge von zwei oder mehr Zeichen, wobei das erste immer den Zeichencode 27 (= ASCII-Code für Escape) hat.

Eben diese Escape-Sequenzen werden auch vom Betriebssystem des Atari ST verstanden (VT52-Emulator). Soll zum Beispiel der Cursor in die linke obere Ecke gesetzt werden, sendet man an den Bildschirm *ESC H*:

```
PRINT CHR$(27);"H";
```

Oder soll an der momentanen Cursor-Position eine Zeile eingefügt werden, sendet man einfach *ESC L*:

```
PRINT CHR$(27);"L";
```

Der untere Teil des Bildschirmes wird dann automatisch verschoben. Eine Übersicht der möglichen VT52-Funktionen gibt die Tabelle.

Das Konzept

Auch wenn der Emulator die meiste Bildschirmarbeit erledigt, geht es doch nicht ganz ohne Verwaltungskram. *ED_IT* verwendet hierfür einige Variablen, in denen die aktuelle Zeilennummer (*ZNR*), die Zeilennummer der obersten Bildschirmzeile (*TOPLINE*) und die Position des Cursors innerhalb einer Zeile (*ZEIGER*) vermerkt sind (siehe auch Bild 1). Die Zeileninhalte liegen in dem Stringarray *ZEILEN$()*. Bei jeder Tastenaktion werden nun diese Variablen aktualisiert. Beispiel: mit jeder Eingabe eines Textzeichens wird *ZEIGER* um eins erhöht, das Zeichen an *ZEILEN$()* angehängt und der Cursor weiterbewegt. Wird eine Cursor-Taste (z.B. hoch) bedient, wird *ZNR* verringert; befindet sich der Cursor am oberen Bildschirmrand, wird *TOPLINE* zurückgezählt, der Bildschirm heruntergescrollt und die oberste Zeile neu eingetragen. Die Unterscheidung, ob eine Zeichen- oder Steuertaste gedrückt wurde, erfolgt über sogenannte Scan-Codes. Bei Tastendruck wird von der Tastatur ein 32-Bit-Wert mit der Tastenkombination geliefert, der wie folgt aufgebaut ist:

Bits 0- 7	ASCII-Code der Taste
Bits 16-23	SCAN-Code der Taste
Bits 24-31	Umschalttastenstatus

Durch Ausmaskieren mit der AND-Funktion werden die benötigten Bits isoliert (siehe auch DEFFN in der Prozedur EDI_-INIT). So kann man sich alle benötigten Informationen über das, was auf der Tastatur los ist, herausholen.

Die Initialisierung

In der Prozedur *EDI-INIT* werden die Konstanten für Bildschirmhöhe, Bildschirmbreite usw. eingelesen sowie der „Textspeicher" *ZEILEN$()* dimensioniert. Auf diese Weise ist eine Anpassung an individuelle Bedürfnisse kein Problem. Die Funktionsdefinitionen *ASCII(x)*, *FTASTE(x)* und *SCAN(x)* filtern die zur Steuerung benötigten Bits aus dem Wert, der vom Tastaturprozessor gesendet wird. Die neue Bildschirmhöhe wird dem System über eine LINE-A (*L~a*)-Variable mitgeteilt. In den Line-A-Variablen stehen Informationen, die GEM und TOS für Bildschirmaktionen usw. dringend benötigen. Also Vorsicht beim Experimentieren! Versehentliche Änderungen in diesem Bereich können einen Absturz, mindestens aber merkwürdiges und unberechenbares Verhalten des Rechners zur Folge haben (Sie kennen sicher das Naturgesetz, nach dem sich der Rechner aufhängt, bevor

man die Arbeit von Stunden abgespeichert hat). Um den Ursprungszustand bei Verlassen des Editors wieder herstellen zu können, wird die Variable *OLD_ZEILEN* als Zwischenspeicher genutzt. Zum Schluß der Vorbereitungen wird der Cursor eingeschaltet und in die linke obere Ecke gesetzt. Die Sequenz *ESC E* löscht nur den eingestellten Bildschirmbereich. Auf diese Weise kann man unterhalb des scrollfähigen Teiles Help-Texte, Bedienungshinweise oder Programminformationen anzeigen, ohne daß diese bei Anwendung des Editors verschoben werden. Grafikfunktionen und Anzeigen mit *TEXT x,y,X$* sind in diesem Bereich unbeschränkt benutzbar, so daß laufende Änderungen kein Problem sind.

Der Editor

Zunächst langweilt sich der Rechner in einer Schleife. Die Hilfsvariable *X* hat solange den Wert Null, bis die Tastatur den oben angesprochenen 32-Bit-Wert liefert. Anschließend wird der zugehörige ASCII-Code herausgezogen und in der Variablen *TASTE* gespeichert. Sollte *TASTE* den Wert Null haben, war es eine Sondertaste (Up, Down, Help, usw.); also wird nun der SCAN-Code ermittelt und anhand der SELECT-CASE-Anweisung verzweigt. Wurde beispielsweise die Cursor-Down-Taste (SCAN 80) gedrückt, wird die Zeilennummer *ZNR* um eins erhöht, *ZEIGER* auf den Zeilenanfang der nächsten Zeile gesetzt und geprüft, ob gescrollt werden muß. Dies ist dann der Fall, wenn sich der Cursor in der untersten Zeile des eingestellten Bildschirmbereiches befindet; also mit *PRINT* einen Upscroll erzwingen und die *TOPLINE* um eins erhöhen. Befand sich aber der Cursor oberhalb der Unterkante, wird er mit *ESC B* um eine Zeile nach unten bewegt. RETURN und Cursor-Up funktionieren entsprechend. *ED_IT* merkt sich immer die höchste Zeilennummer in der Variablen *LASTLINE*, damit ein Durchfahren des Textes nur bis zur letzten Zeile möglich ist. Wollen Sie den Editor um weitere Funktionen erweitern, brauchen Sie nur den SCAN-Code der Taste zu ermitteln und eine weitere „CASE-xy-Schachtel" hinzuzufügen, in der dann Ihre Funktionen liegen.

Der Editor kann mittels UNDO-Taste wieder verlassen werden. Wie schon oben erwähnt, ist es wichtig, die Line-A-Variable Int{*L~a-42*} auf den alten Wert zurückzusetzen, um böse Überraschungen zu vermeiden.

Literatur:
[1] Handbuch GFA-BASIC 3.0, GFA Systemtechnik GmbH
[2] Brückmann/Englisch/Gerits: ST-Intern, DATA BECKER GmbH

Übersicht der VT52-Escape-Sequenzen

ESC „A"	Cursor hoch ohne Scrolling
ESC „B"	Cursor runter ohne Scrolling
ESC „C"	Cursor rechts
ESC „D"	Cursor links
ESC „E"	Clear Home (löscht Bildschirm)
ESC „H"	Cursor Home (linke obere Ecke)
ESC „I"	Cursor hoch mit Scrolling
ESC „J"	löscht den Rest des Bildschirmes ab Cursor-Position.
ESC „K"	löscht den Rest der Zeile, in der sich der Cursor befindet.
ESC „L"	Zeile an Cursor-Position einfügen. Unterer Bildschirmteil wird um eine Zeile nach unten verschoben.
ESC „M"	entfernt die Zeile, in der der Cursor steht, und zieht den Rest des Bildschirmes hoch.
ESC „Y"	Positionieren des Cursors. Die folgenden zwei Zeichen werden als Zeile und Spalte interpretiert, wobei jeweils 32 addiert werden muß. Soll der Cursor beispielsweise auf Zeile 8, Spalte 12 gesetzt werden, schreibt man: PRINT CHR$(27);"Y";CHR$(8+32);CHR$(12+32);
ESC „b"	Einstellung der Schriftfarbe. Je nach Modus 2, 4 oder 16 Farben (zB. schwarz: ESC „b1")
ESC „c"	Einstellung der Hintergrundfarbe
ESC „d"	Löschen des Bildschirmes bis Cursor-Position
ESC „e"	Einschalten des Cursors
ESC „f"	Abschalten des Cursors
ESC „j"	speichert die augenblickliche Cursor-Position
ESC „k"	setzt Cursor wieder auf gespeicherte Position.
ESC „l"	löscht nur die aktuelle Zeile.
ESC „o"	löscht von Zeilenanfang bis Cursor-Position
ESC „p"	inverse Schrift einschalten
ESC „q"	inverse Schrift abschalten
ESC „v"	Zeilenumbruch an. Bei Erreichen des Zeilenendes wird Ausgabe in der nächsten Zeile fortgesetzt.
ESC „w"	Zeilenumbruch aus

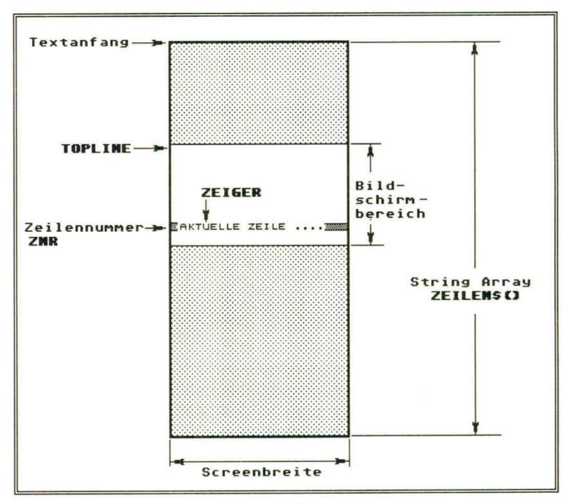


Bild 1: Zusammenhang zwischen Variablen und Textspeicher

```
 1:  '
 2:  ' ED_IT.GFA     von Rolf Stelljes
 3:  ' Mini-Texteditor
 4:  ' GFA 3.07
 5:  ' (c) MAXON Computer 1991
 6:  '
 7:  edi_init
 8:  editor
 9:  '
10:  PROCEDURE edi_init     ! Vorbereitung
                              der Editorfunktionen
11:    '
12:    ' Alle Numerischen Variablen Integer-Format %
13:    ' Die Bildschirmsteuerungen erfolgen über
         VT52-Sequenzen
14:    '
15:    DIM zeilen$(500)         ! Array für
                              Textzeilen (Größe abhängig
16:    '                        ! von Verwendungszweck)
17:    DEFFN ascii(x%)=x% AND 255 ! Ermittelt
                                    ASCII-Code einer Taste
18:    DEFFN ftaste(x%)=@scan(x%)-58    ! Ermittelt F-
                                         Tasten-Nummer  →
```

VORTEX DATAJET UND ATONCE-PLUS

Das leistungsstarke Doppel: DataJet-AT für DM 1.498,--*
40 MB Festplatte plus AT-Emulator für Atari ST/Mega ST

vortex DataJet-AT das Kombinations-Produkt aus Festplatten-Subsystem vortex DataJetX40 und AT-Emulator vortex ATonce-Plus. Jetzt noch mehr Leistung durch vortex ATonce-Plus mit 16 MHz und Norton 8.0. Durch dieses leistungsstarke Doppel wird Ihr Atari ST Computer zu einem "Business-like" AT-kompatiblen Computer.

vortex ATonce-Plus: ■ Norton 8.0 ■ 80286 CPU ■ 16 MHz ■ Mit EGA-/VGA-Monochrom-Graphik[1] ■ FontMaster (Möglichkeit den eigenen Zeichensatz zu editieren) ■ vortex HyperSwitch Accessory (Schnelles Umschalten zwischen AT-Mode und Atari-Mode) ■ Windows 3.0 läuft uneingeschränkt im Protected Mode ■ Expanded/Extended Memory-Verwaltung ■ Hohe AT-Kompatibilität bei Fest-/Wechselplatten, Floppy-Laufwerken, Grafik-Modi, serielle Schnittstelle

vortex DataJetX40: ■ 40 MB Speicherkapazität ■ SCSI-Laufwerk ■ Hardware-Schreibschutz ■ Motor- und Lüfter-Steuerung ■ Technische Standards VDE, VDI und GS ■ AHDI3.0-Kompatibilität
vortex DataJet-AT: ■ DataJetX40 an Atari anschließen ■ ATonce-Plus in Atari einbauen/stecken ■ AT-Kompatibilität zum Preis von DM 1.498,- (*Unverbindlicher empfohlener Endverbraucherpreis) ■ Software-Update-Service ■ Täglich Kundenberatung



[1] soweit es die Fähigkeiten des Atari ST/Mega ST zulassen. Alle Firmen- und Produktnamen sind Warenzeichen der jeweiligen Inhaber und urheberrechtlich geschützt.

VORTEX COMPUTERSYSTEME GMBH . FALTERSTRASSE 51-53 . D-7101 FLEIN . TELEFON 07131 / 59 72-0
DTZ DATATRADE AG . LANDSTRASSE 1 . CH-5415 RIEDEN/ BADEN . TELEFON 056/ 82 18 80

```
19:   DEFFN scan(x%)=((x% AND &HFF0000)/65536)
                                  ! Ermittelt Scan-Code
20:   scr_ho%=10                  ! Bildschirmhöhe
21:   scr_br%=30                  ! Bildschirmbreite
22:   znr%=1                      ! Erste Z-Nr.
23:   zeiger%=1                   ! Zeilenanfang
24:   tabs%=8                     ! Tabulatorschritte
25:   topline%=1                  ! Oberste Zeile
                                    des Bildschirms
26:   raus!=FALSE                 ! Merker zurück-
                                    setzen
27:   WHILE INKEY$<>""            ! Tastaturpuffer
                                    leeren
28:   WEND
29:   '
30:   old_zeilen%=INT{L~A-42}     ! Maximale Anzahl
                                    Zeilen merken
31:   INT{L~A-42}=scr_ho%-1       ! Neue max Zeilen-
                                    anzahl setzen
32:   '
33:   PRINT CHR$(27);"e";         ! Cursor ein
34:   PRINT CHR$(27);"H";         ! Cursor in linke
                                    obere Ecke
35: RETURN
36: PROCEDURE editor              ! Texteditor mit
                                  Scrolling, Ins, Del.
37:   REPEAT                      ! Hauptschleife
38:     REPEAT                    ! Warten auf
                                    Tastendruck
39:       KEYTEST x%              ! und Tasten-
                                    code merken
40:     UNTIL x%>0
41:     taste%=@ascii(x%)         ! ASCII-Code
                                    ausfiltern
42:     IF taste%=0               ! Falls Steuertaste
                                    bedient
43:       SELECT @scan(x%)        ! Scan-Code
                                    ermitteln
44:       CASE 82                 ! ** Insert **
45:         PRINT CHR$(27);"L";  ! Leerzeile an
                                  CrsPosition einfügen
46:         INSERT zeilen$(znr%)="" ! Platz in
                                    Array einfügen
47:         INC lastline%           ! Anzahl der
                                    Zeilen erhöhen
48:         zeiger%=1               ! Crs an
                                    linken Rand
49:       CASE 80                   ! ** Down **
50:         IF znr%<lastline%       ! Runter nur
                                    bis letzte Zeile
51:           INC znr%              ! Nächste
                          Zeilennummer (max Lastline)
52:           zeiger%=1             ! Zeilenanfang
53:           IF CRSLIN=>scr_ho%    ! Falls
                               Unterkante Bildschirm
                               erreicht
54:             INC topline%        ! Topline
                                    verschieben
55:             PRINT               ! Bildschirm
                                    hochscrollen
56:           ELSE                  ! ansonsten nur
57:             PRINT CHR$(27);"B"; ! Crs down
58:           ENDIF
59:           PRINT AT(1,CRSLIN);zeilen$(znr%);
                AT(1,CRSLIN); ! Zeileninhalt zeigen
60:         ENDIF
61:       CASE 72                   ! ** Up **
62:         znr%=MAX(1,znr%-1) ! Vorige Zeilennummer
                                   (minimal 1)
63:         zeiger%=1               ! Zeilenanfang
64:         IF CRSLIN=1             ! Falls oberer
                                    Bildschirmrand
65:           IF topline%>1     ! Hochscrollen
                             nur bis Oberkante Text
66:             PRINT CHR$(27);"I";  ! Crs up mit
                                    Scrolling down
67:           ENDIF
68:           topline%=MAX(1,topline%-1)! Topline
                                    verschieben
69:         ELSE                    ! Falls nicht
                                Bildschirmoberkante
70:           PRINT CHR$(27);"I"; ! Crs hoch
71:         ENDIF
72:         PRINT AT(1,CRSLIN);zeilen$(znr%);
              AT(1,CRSLIN); ! Zeileninhalt
                                anzeigen
73:       CASE 77                   ! ** Right **
74:         IF zeiger%<LEN(zeilen$(znr%))+1 ! Zeiger
                              maximal bis Zeilenende
75:           PRINT CHR$(27);"C";           ! Crs
                                             rechts
76:           zeiger%=MIN(zeiger%+1,scr_br%) !
                          Zeiger nach rechts verschieben
77:         ENDIF                          !
78:       CASE 75                          ! ** Left **
79:         PRINT CHR$(27);"D";           ! Crs links
80:         zeiger%=MAX(zeiger%-1,1)  ! Zeiger nach
                                  links verschieben
81:       CASE 97                          ! ** Undo **
82:         INT{L~A-42}=old_zeilen%    ! Alten
                            Zustand wieder herstellen
83:         PRINT CHR$(27);"f";  ! Crs ausschalten
84:         raus!=TRUE               ! Fertig-Flag
                                       setzen
85:       ENDSELECT                  ! Ende
                                 Steuertastenverteiler
86:     ELSE
87:       x$=zeilen$(znr%)          ! Aktuelle Zeile
                                in Hilfsstring legen
88:       y$=x$
89:       SELECT taste%             ! Falls ASCII-
                                      Zeichen
90:       CASE 32 TO 126,129,132,142,148,153,154,158
                                 ! ** Zeichentaste
91:         IF zeiger%<scr_br%
92:           x%=LEN(x$)-zeiger%+1  ! Restlänge
                                  der Zeile ab Zeiger
93:           x$=LEFT$(x$,zeiger%-1)+CHR$(taste%)
                                 ! Neues Zeichen vor
                                   Zeiger einfügen
94:           IF x%>0               ! Falls Rest
                                      vorhanden
95:             x$=x$+RIGHT$(y$,x%)  ! Restzeile
                                       anhängen
96:           ENDIF
97:           zeiger%=MIN(zeiger%+1,scr_br%)! Zeiger
                                    weiter bewegen
98:           PRINT AT(1,CRSLIN);x$;  ! Zeile
                                       anzeigen
99:           PRINT AT(zeiger%,CRSLIN);! Crs
                                      positionieren
100:          zeilen$(znr%)=x$         ! Neue Zeile
                                       merken
101:        ENDIF
102:      CASE 9                        ! ** Tab **
103:        x%=((zeiger%+tabs%) DIV tabs%)*tabs%
                    ! Nächste Tab-Position berechnen
104:        zeiger%=MAX(1,MIN(LEN(zeilen$(znr%))+1,
             x%))  ! Zeiger positionieren
105:        PRINT AT(zeiger%,CRSLIN); ! Crs posit.
106:      CASE 13                       ! ** Return **
107:        INC znr%                    ! Neue
                                      Zeilennummer
108:        IF CRSLIN=>scr_ho%          ! Falls
                         Unterkante Bildschirm erreicht
109:          INC topline%              ! Topline
                                      verschieben
110:          PRINT                     ! Bildschirm
                                      hochscrollen
111:        ELSE                        ! ansonsten
                                      nur...
112:          PRINT CHR$(27);"B";      !   Crs runter
113:        ENDIF                       !
114:        PRINT AT(1,CRSLIN);zeilen$(znr%);AT(1,
             CRSLIN);          ! Zeileninhalt anzeigen
115:        zeiger%=1          ! Zeiger auf
                              Zeilenanfang setzen
116:      CASE 25              ! ** Ctl Y **
                              (Zeile löschen)
117:        PRINT CHR$(27);"M";   ! Zeile auf
                       Bildschirm löschen mit upscroll
118:        DEC lastline%          ! Anpassung
                                  Zeilenanzahl
119:        DELETE zeilen$(znr%)  ! Zeile aus
                                  Array entfernen
120:        zeiger%=1              ! Zeiger auf
                                  Zeilenanfang
121:      CASE 27                  ! ** Escape **
                              (Zeileninhalt löschen)
122:        PRINT CHR$(27);"l";   ! Zeile auf
                                Bildschirm löschen
123:        zeilen$(znr%)=""      ! Inhalt löschen
124:        zeiger%=1              ! Zeiger auf
                                  Zeilenanfang
```

→

```
125:          CASE 8,127                 ! ** Delete/
                         Backspace ** (Zeichen löschen)
126:             x$=zeilen$(znr%)        ! Aktuelle
                         Zeile an Hilfsstring übergeben
127:             x%=LEN(x$)              ! Länge merken
128:             IF taste%=8             ! Falls Backspace-
                                           Taste
129:               zeiger%=MAX(zeiger%-1,1)  ! voriges
                                   Zeichen anvisieren
130:             ENDIF
131:             IF zeiger%<=x%          ! Falls Zeichen
                                      innerhalb Zeile
132:               zeilen$(znr%)=LEFT$(x$,zeiger%-1)+
                    RIGHT$(x$,x%-zeiger%) ! dieses
                                            isolieren
133:               PRINT AT(1,CRSLIN);zeilen$(znr%);" ";
                                   ! Zeile neu anzeigen
134:               PRINT AT(zeiger%,CRSLIN);! Crs
                                         positionieren
135:             ENDIF
136:           ENDSELECT                  ! Ende ASCII-
                                       Tastenverteiler
137:         ENDIF
138:         lastline%=MAX(lastline%,znr%)  ! Höchste
                                  Zeilennummer merken
139:       UNTIL raus!
140:     RETURN
```

DER BILDVERBIEGER

Thomas Schweitzer

ICH BIN SICHER, SIE KENNEN UND LIEBEN SIE ALLE, DIE VERBOGENEN BILDER AUS DEM COMPUTER, ODER GENAUER GESAGT, DIE RECHTECKIGEN BILDKLÖTZE, DIE EINEM SCHNUCKELIGEN VIERECK ANGEPASST WERDEN. WER DER WARTEZEITEN DER DEGAS-FUNKTION DISTORT ÜBERDRÜSSIG IST, SOLLTE SICH MAL ANSEHEN, WIE EINFACH UND SCHNELL DAS MIT MEINEM VERFORMUNGSALGORITHMUS LÄUFT.

Um die Spannung noch ein bißchen zu heben, möchte ich gleich darauf hinweisen, daß dieses Verfahren noch mehr kann als simple Vierecksverformung. Wie wär's z.B. mit einer Ellipsenscheibe, einer Tonne, einem Kegel, einer Lochscheibe usw.?

Für all diejenigen, die jetzt schon ganz kribbelig geworden sind, möchte ich zunächst mit der Vierecksverformung beginnen.

Das Gummiexperiment

Machen Sie also Ihre Augen zu und stellen Sie sich ein Gummirechteck (unser Bild) vor, auf das ein Raster gezeichnet wurde. Wenn wir jetzt gemeinsam an den Ecken des Gummis ziehen, erhalten wir ein Viereck, das nicht mehr ganz so rechteckig ist (vielleicht könnte der Herr an der linken oberen Ecke noch etwas stärker ziehen, dann wird's deutlicher). Was wir nun sehen, bringt uns sofort auf eine Idee: denn, o Wunder, alle Rasterlinien bleiben schön gerade und haben immer noch dieselben Anfangs- und Endpunkte, die nur etwas weiter auseinandergerückt sind, aber noch im selben Abstandsverhältnis zueinander stehen, wie die neuen zu den alten Seiten. So, jetzt können Sie die Augen wieder aufmachen und auf Bild 1 schauen. Stellen Sie sich das Raster nun noch ganz dicht und bunt vor, und schon haben Sie ein Bild.

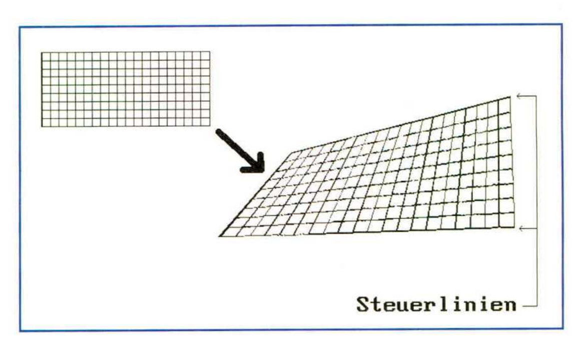


Bild 1: Das Gummiexperiment

Die Steuerlinien

Von dieser Idee erleuchtet, bestimmen wir nun zwei 'Steuerlinien', sagen wir mal die unterste und oberste Linie des ursprünglichen Rechteckes, die ja immer noch existieren, sich aber irgendwo anders rumräkeln. Alle Punkte der Steuerlinien legen wir über ihre Koordinatenpaare in den zweidimensionalen Feldern *xlist[][]* und *ylist[][]* ab. *xlist[1][15]* enthält z.B. die x-Koordinate des 16. Punktes in der zweiten Steuerlinie. Die entsprechenden Punktepaare (komme ich gleich zu) der beiden (horizontalen) Steuerlinien müssen nun nur noch durch (vertikale) Geraden verbunden werden, die die Farbinformationen des Ursprungsbildes (auch nur ein Haufen vertikaler Linien) enthalten. (Vertikal und horizontal stehen in Klammern, da sie im Quellrechteck eben so angeordnet waren, dies aber im neuen Viereck nicht so sein muß.)

Linien auf der Streckbank

Aus der vorletzten Reihe höre ich gerade den bösen Zwischenruf, die (vertikalen) Linien seien doch wohl nicht mehr genauso lang wie vorher. Recht hat er! Bevor wir eine (vertikale) Strecke zwischen zwei Punkte der beiden (horizontalen) Steuerlinien spannen, müssen wir die vertikale Strecke des Ursprungsbildes entsprechend dem Abstand dieser Punkte strecken bzw. stauchen. Dies geschieht in der Funktion *makelist*, die die Punkte in der Spalte *x* zwischen den Begrenzern *y1* und *y2* mit *len* Farbwerten ins Feld *colors[]* (global) kopiert. Die Werte des Feldes *colors[]* repräsentieren also die Farben einer vertikalen Linie im Quellbild, wobei diese auf eine Länge von *len* Punkten gestreckt bzw. gestaucht wurde.

Die Farben werden mit der Funktion *get_pix* ermittelt, die auch auf eine Bildschirmseite zugreifen kann, die weder physikalisch noch logisch beim XBIOS angemeldet ist. Der

Vorteil liegt darin, daß nicht dauernd zwischen dem Arbeitsschirm, auf den gezeichnet wird, und dem Originalbild umgeschaltet werden muß.

Linienzauber

Schon die ganze Zeit war die Rede von Steuerlinien und eingespannten Linien, doch wie wir diese schnell berechnen, wollte uns keiner sagen, bis Herr Bresenham mal auf die Idee kam, seinen superschnellen Integer-Bresenham-Algorithmus [1][2] vorzustellen. Dieses Verfahren zeichnet sich dadurch aus, daß es nicht mit langsamen float-Werten rumfummelt, sondern durch einfache Integeraddition bzw. -subtraktion zum Ziel kommt. Nun kann man Rasterlinien ja auf zwei verschiedene Arten zeichnen (Bild 2). Die Punkte können sich bei Änderung der Richtung im Linienverlauf überlappen (a) oder auch nicht (b). Die Linien des Typs a) sehen als Einzelgänger zwar etwas pummelig aus, aber im Rudel haben sie besondere Qualitäten. Anhand des Parameters *mode* (= dick bzw. duenn) der Funktion *line* werden die Linientypen gewählt.

Was dem Käse recht, ist bei uns ganz schlecht: die Löcher. Wie oben angedeutet, möchten wir ja viele Linien nebeneinander zeichnen, auf daß sich unser Bild wieder ergebe. Wie wir aber in Bild 2 sehen, sind Linien des Typs b) dafür nicht sonderlich geeignet, da sie häßliche Löcher im Bild hinterlassen. Nach etwas Fummelei mit sich kreuz und quer überlagernden Linien kam ich dann auf die Idee mit den Linien vom Typ a), die im Verbund selbst in Extremlagen äußerst wasserdicht sind (bei Degas' *Distort* ist dies nicht immer gegeben).

Steuerlinien - die zweite

Wer vorhin gut aufgepaßt hat, weiß jetzt immer noch nicht, was entsprechende Punktepaare der beiden Steuerlinien sind. Wir haben also schon festgestellt, daß die Steuerlinien unterschiedlich lang sein können. Ist z.B. die erste Steuerlinie dreimal so lang wie die zweite, müssen also je drei Linien von drei nebeneinanderliegenden Punkten auf der ersten Steuerlinie zu einem Punkt auf der zweiten Steuerlinie gezeichnet werden. Die Koordinaten nehmen wir dabei natürlich aus unseren Koordinatenfeldern *xlist* und *ylist*. *xlist[0][..]* und *ylist[0][..]* werden also mit einer Schrittweite von 1 und *xlist[1][..]* und *ylist[1][..]* mit 1/3 durchlaufen. Otto Normalverbraucher würde hier natürlich wieder seine langsame float-Arithmetik bemühen. Wir haben jedoch nicht so viel Zeit und machen es anders - mit der Integerarithmetik.

Fließende Integer

Nehmen wir einmal an, eine Genauigkeit von 3 Nachkommastellen würde für unsere Zwecke ausreichen. Dann könnten wir unsere Float-Zahlen 1 und 0.333 doch auch als Integer 1000 und 333 schreiben und vor der Indizierung im Feld wieder durch 1000 dividieren (Ganzzahldivision). So würde aus 333,666,999 die 0, aus 1332,1665,1998 die 1 usw. Und wir wollten ja erreichen, daß die ersten drei Linien zum Punkt 0 der zweiten Steuerlinie laufen, die nächsten drei zum zweiten Punkt usw. Nun kommt unser 68000er bei Multiplikationen bzw. Divisionen mit Zehnerpotenzen aber ganz schön ins Schwitzen. Mit Zweierpotenzen steht er sich jedoch schon viel besser, da er z.B. für eine Multiplikation mit 4 den Wert in seinen Registern nur um 2 Stellen nach links schubsen muß, und das macht er mit links.

Im Programm finden wir deshalb die Konstante *BITMOVE* (kein Bierwagen), die für die Genauigkeit unserer Berechnungen zuständig ist. Mit einem Wert von 12 erreichen wir also eine ca. 3,5stellige Genauigkeit (2^12 = 4096) hinter dem Komma.

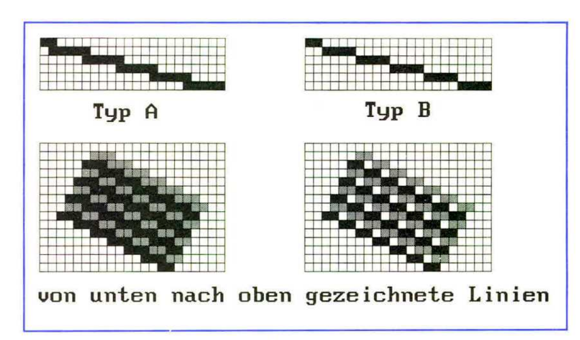


Bild 2: Linientypen alleine und im Rudel

So läuft der Hase

Schauen wir uns jetzt also einmal die Funktion *verform*, das Herzstück des Programms, an. Sie wird mit den folgenden Parametern aufgerufen: *x0* und *y0* sind Offsets, die erst bei Tonnen, Kegeln und Lochscheiben, besonders in Hinblick auf Animationserstellung (ich arbeite daran), interessant werden. Sie können ja mal ein bißchen damit rumspielen oder sie einfach auf 0 setzen. *x1,y1,x2,y2* sind die linke obere und rechte untere Ecke des Ursprungsrechtecks. *x3,y3-x6,y6* beschreiben das neue Viereck von der linken oberen Ecke im Uhrzeigersinn ausgehend. Kommen wir nun zum Eingemachten.

Nachdem wir die Maus vorsorglich zu Bett gebracht haben, werden die beiden Listen mit den Punktkoordinaten der Steuerlinien mit Linientyp b) (duenn) erzeugt. Dann ermitteln wir die längste der beiden Linien und berechnen aus ihr die Schrittweite in den Koordinatenlisten (eine ist immer 1 * 2^*BITMOVE*). Das Originalbild muß horizontal natürlich auch in *laengst* Schritten abgetastet werden (*steporig*).

pos1 und *pos2* sind die Indizes in den Koordinatenfeldern, *oz* der Spaltenzähler im Originalbild. *pos1g*, *pos2g* und *ozg* sind die arithmetisch genaueren Entsprechungen dazu.

Solange nun das Ende einer Steuerlinie nicht erreicht ist, wird mit *makelist* das colors-Feld gefüllt und die (vertikalen) Linien zwischen den Steuerlinien gezeichnet [*mode* = dick (Typ a)]. Der globale Zähler *z* und die Funktion *plot* sorgen dafür, daß die Punkte der Linie die richtige Farbe erhalten, indem *colors[ ]* ausgelesen wird. In *plot* wird zudem noch die Variable *transparent* ausgewertet. Hat sie den Wert TRUE, werden nur Punkte mit einer Farbe größer als 0 gesetzt, d.h. der Hintergrund wird nicht überdeckt.Da in diesem Modus i.a. auch weniger Punkte gesetzt werden, erhöht sich hier die Zeichengeschwindigkeit.

Optimierungen,...

...die möglich sind, aber nicht viel verbessern: um ein Bild auf die volle Bildschirmgröße von 640 * 400 Punkten zu verformen, benötigt das Programm in der vorliegenden Version ca. 1 Minute und 3 Sekunden (nicht transparent). Eine Optimierung könnte bei der häufigst frequentierten Routine *get_pix* ansetzen.

In Assembler formuliert schafft sie einen unglaublichen Vorsprung von 4 Sekunden. Wenn wir die Farbwerte in einem zweidimensionalen Byte-Feld ablegen, gewinnen wir ca. 3 Sekunden. Dies ist, wenn überhaupt, erst dann lohnend, wenn der gleiche Bildausschnitt oft bearbeitet wird (z.B. Animationssequenzen berechnen). Nun, alle diese Optimierungen reißen uns nicht vom Hocker; ist ja auch so schon schnell genug, oder?

Zur schmackhaften Zubereitung des Programmes benötigen wir einen Atari ST, einen Turbo C-Compiler (Version 2.0) und

etwas Zeit. Nachdem wir die Programmzeilen sorgfältig und liebevoll eingetippt haben, schieben wir das ganze in den Compiler und lassen es ca. 16 Sekunden bei 8 MHz mit dem Default-Projekt garen. Daß dabei 3 Warnungen erscheinen, übergehen wir mit einem Anflug von Heiterkeit.

Das hier veröffentlichte Programm ist eigentlich nur eine Fuzzidemoversion, mit der Sie nur ein Bild laden können. Dann müssen das Quell- und das Zielrechteck angegeben werden. Nachdem das Bild neu berechnet wurde, können Sie mit *ESC* abbrechen oder auch von vorne anfangen. Das ganze arbeitet übrigens auflösungsunabhängig (abgesehen von Grafikkarten, die kein Atari-Format verwenden).

Wer seine Bilder drehen möchte (und dabei gerne 7mal schneller ist als Degas), braucht die Eckpunkte seines Rechtecks nur der Transformation

$$xneu=(x-xm)*\cos(phi)-(y-ym)*\sin(phi)+xm$$

$$yneu=(y-ym)*\cos(phi)+(x-xm)*\sin(phi)+ym$$

zu unterwerfen, wobei *xm* und *ym* jeweils die Rotationsmittelpunkte sind.

„Wo bleiben denn nun die Kegel und Tonnen?“, werden Sie jetzt schreien. Dann stellen Sie sich einfach mal zwei Steuerkreise anstelle von Steuerlinien vor oder einen Steuerkreis und einen Fixpunkt.



Literatur:
[1] Luther / Ohsmann: Mathematische Grundlagen der Computergraphik, Vieweg 1988
[2] Plenge : Das Supergrafikbuch zum Atari ST, Data Becker 1987

```
  1:  /****************************************************
  2:
  3:      Programm Bildverbieger
  4:
  5:      entwickelt von Thomas Schweitzer Oktober '90
  6:      mit Turbo C 2.0
  7:
  8:      Copyright by MAXON Computer GmbH
  9:
 10:  ***************************************************/
 11:
 12:  #include <tos.h>
 13:  #include <stdio.h>
 14:  #include <stdlib.h>
 15:  #include <linea.h>
 16:  #include <ext.h>
 17:  #include <string.h>
 18:  #include <vdi.h>
 19:
 20:  #define duenn 0  /* Linienzeichenmodi */
 21:  #define dick  1
 22:  #define MAXPTS 2100 /* Max. Länge einer Linie */
 23:
 24:  #define BITMOVE 12
 25:          /* Genauigkeit der Intergerarithmetik */
 26:
 27:  #define ABS(x)  (((x)<0) ? -(x) : (x))
 28:  #define max(x,y)  (((x)<(y)) ? (y) : (x))
 29:
 30:  typedef enum
 31:   {FALSE,
 32:    TRUE  }
 33:  boolean;
 34:
 35:  int line(int x1, int y1, int x2, int y2,
 36:           void (*action)(int,int,int),int mode);
 37:  void verform(int x0, int y0,
 38:               int x1, int y1, int x2, int y2,
 39:               int x3, int y3, int x4, int y4,
 40:               int x5, int y5, int x6, int y6);
 41:
 42:  void list1(int x,int y, int dummy);
 43:  void list2(int x,int y, int dummy);
 44:
 45:  int  get_pix(int x,int y,void *logbase);
 46:  void plot ( int x, int y, int col);
 47:
 48:  void makelist(int len, int x,
 49:                int y0, int y1, int y2);
 50:
 51:  size_t BildLaden(void *base, char *Name);
 52:  void getSource(int *x1,int *y1,int *x2,int *y2);
 53:  void getDest   (int *x3,int *y3,int *x4,int *y4,
 54:                  int *x5,int *y5,int *x6,int *y6);
 55:
 56:  void mymain (void);
 57:  void close_vwork( void );
 58:  void *ZweiterSchirm (void);
 59:  boolean open_vwork( void );
 60:
 61:  int z,     /* Punktezähler in list1 und list2 */
 62:      ylist[2][MAXPTS],
 63:      xlist[2][MAXPTS];
 64:
 65:  char colors[MAXPTS];
 66:
 67:  int  PDBitmove;
 68:
 69:  void *GemScreen,
 70:       *Original,  /* Bildschirmadressen */
 71:       *Arbeit;
 72:
 73:  boolean transparent = TRUE;
 74:
 75:  void main( void )
 76:  {
 77:     if ( open_vwork( ) == TRUE ) {
 78:       mymain( );
 79:
 80:       close_vwork( );
 81:       exit ( 0 ); }
 82:     else {
 83:       fprintf(stderr,"Fehler bei der"
 84:                      "Programminitialisierung!" );
 85:        exit ( -1 );
 86:     }
 87:  }
 88:
 89:  /****************************************************
 90:
 91:   boolean open_vwork( void )
 92:
 93:   Öffnet die Arbeitsstation und initialisiert
 94:   dabei folgende Variablen
 95:
 96:   int MAXX         Auflösung in x-Richtung
 97:   int MAXY         Auflösung in y-Richtung
 98:   int BytesZeile   Bytes pro Bildschirmzeile
 99:   int Planes       Anzahl der Farbebenen
100:   int Colors       Anzahl der Farben
101:
102:   size_t PICLEN    Bildlänge in Bytes
103:   size_t LPICLEN   Bildlänge in Langworten
104:
105:   int work_in[12]
106:   int work_out[57] VDI-Arbeitsfelder
107:
108:   int handle       Bildschirmhandle
109:   int phys_handle Bildschirmhandle
110:
111:   int gl_hchar
112:       gl_wchar
113:       gl_hbox
114:       gl_wbox      Höhe und Breite der Systemfonts
115:   int gl_apid      Applikationsnummer
116:
117:   -> nix
118:   <- kein Fehler  TRUE
119:   <- Fehler       FALSE
120:
121:  ***************************************************/
122:
123:  int MAXX,MAXY,
124:      BytesZeile, /* Bytes pro Bildschirmzeile */
125:      Planes,     /* Anzahl der Farbebenen     */
126:      Colors;     /* Anzahl der Farben         */
127:
128:  size_t PICLEN,  /* Bildlänge in Bytes        */
129:         LPICLEN; /* Bildlänge in Langworten   */
130:
131:  int work_in[12],
132:      work_out[57];
133:
134:  int handle,
```

→

```
135:       phys_handle;
136:
137:  int gl_hchar,
138:      gl_wchar,
139:      gl_hbox,
140:      gl_wbox,
141:      gl_apid;
142:
143:  boolean open_vwork( void )
144:  {
145:    register int i;
146:
147:     if (( gl_apid = appl_init() ) != -1 )
148:     {
149:        for ( i = 1; i < 10; work_in[i++] = 1 );
150:        work_in[10] = 2;
151:        phys_handle = graf_handle( &gl_wchar,
152:                    &gl_hchar, &gl_wbox, &gl_hbox );
153:        work_in[0]  = handle = phys_handle;
154:
155:        v_opnvwk( work_in, &handle, work_out );
156:
157:        MAXX   = work_out[0];
158:        MAXY   = work_out[1];
159:        Colors = work_out[13];
160:
161:        i      = Colors;
162:        Planes = 0;
163:        while (i != 1) {
164:         i /= 2;
165:         Planes++;
166:        }
167:
168:        linea_init();
169:        BytesZeile = Vdiesc->bytes_lin;
170:        PICLEN     = ((size_t) BytesZeile *
171:                      (size_t) (MAXY+1));
172:        LPICLEN    = PICLEN >> 2;
173:        return ( TRUE );
174:     }
175:     else
176:        return ( FALSE );
177:  } /* open_vwork */
178:
179:  void close_vwork( void )
180:  /***************************************************
181:
182:   SchliePen der Arbeitsstation
183:   -> nix
184:   <- nix
185:
186:  ***************************************************/
187:  {
188:     v_clsvwk( handle );
189:
190:     appl_exit( );
191:  }
192:
193:  void mymain(void)
194:  {
195:    char Pfad[40]     = "E:\\*.DOO",
196:         Pfadneu[54] = "\0",
197:         Name[14]    = "\0",
198:         *ptr_to_backslash;
199:
200:    int button,
201:        x1,y1, /* Startrechteck links oben    */
202:        x2,y2, /*       "       rechts unten */
203:        x3,y3, /* Zielviereck links oben      */
204:        x4,y4, /* Zielviereck rechts oben     */
205:        x5,y5, /* Zielviereck rechts unten    */
206:        x6,y6; /* Zielviereck links unten     */
207:
208:    /* Jetzt geht's erst richtig los */
209:    GemScreen = Physbase();
210:    if ((Original = ZweiterSchirm()) == NULL
211:      ||(Arbeit   = ZweiterSchirm()) == NULL)
212:       form_alert(1,"[3][Zu wenig Speicher!]"
213:                    "[Sch...ade]");
214:    else {
215:      PDBitmove = 3-Getrez();
216:      /* Doodlebild selektieren */
217:      fsel_input( Pfad, Name, &button);
218:      if (button == 1) {  /* OK */
219:        strcpy(Pfadneu,Pfad);
220:        ptr_to_backslash = strrchr(Pfadneu, '\\');
221:        strcpy(++ptr_to_backslash,Name);
222:        printf("\nLade %s...",Pfadneu);
223:        if (BildLaden(Original,Pfadneu) != 0){
224:      /* Bild laden und bis Tastendruck anzeigen */
225:          Setscreen(Original,Original,-1);
226:          getch();
227:          Setscreen(GemScreen,GemScreen,-1);
228:
229:          do {
230:            /* Quellblock und ... */
231:            getSource(&x1,&y1,&x2,&y2);
232:            /* ... Zielviereck eingeben */
233:            getDest   (&x3,&y3,&x4,&y4,&x5,&y5,
234:                       &x6,&y6);
235:
236:            /* auf Arbeitschirm verformen und ...*/
237:            Setscreen(Arbeit,Arbeit,-1);
238:            v_clrwk(handle);
239:            verform(0,0,      /* hier geht's rund */
240:                    x1,y1,x2,y2,
241:                    x3,y3,x4,y4,x5,y5,x6,y6);
242:            getch(); /* ... auf Taste warten */
243:             Setscreen(GemScreen,GemScreen,-1);
244:             v_clrwk(handle);
245:             puts("\nESC -> Ich mag nicht mehr, "
246:                   "obwohl's so schön war.\n"
247:                   "sonstwas -> Wow! War ja super "
248:                   "Nochmal bitte."); }
249:          while (getch() != 27);}
250:        /* solange Bilder kaputtmachen bis ESC
251:           gedrückt */
252:        else puts("\nKonnte Bild nicht laden.");
253:      }
254:    }
255:    return;
256:  } /* mymain */
257:
258:  void getSource(int *x1,int *y1,int *x2,int *y2)
259:  {
260:    v_clrwk( handle );
261:    puts("\nHallo User, gib mal das Startrechteck "
262:         "ein.");
263:    printf("\nLinke obere Ecke (x,y)  : ");
264:    scanf("%d,%d",x1,y1);
265:    printf("\nRechte untere Ecke (x,y): ");
266:    scanf("%d,%d",x2,y2);
267:  } /* end getSosse */
268:
269:  void getDest   (int *x3,int *y3,int *x4,int *y4,
270:                  int *x5,int *y5,int *x6,int *y6)
271:  {
272:    v_clrwk( handle );
273:    puts("\n Und jetzt das Zielviereck...");
274:    printf("\nLinke obere Ecke (x,y)  : ");
275:    scanf("%d,%d",x3,y3);
276:    printf("\nRechte obere Ecke (x,y) : ");
277:    scanf("%d,%d",x4,y4);
278:    printf("\nRechte untere Ecke (x,y): ");
279:    scanf("%d,%d",x5,y5);
280:    printf("\nLinke untere Ecke (x,y) : ");
281:    scanf("%d,%d",x6,y6);
282:    printf("\n Vielen Dank, ich stürze jetzt ab.");
283:    getch();
284:  } /* end getDest */
285:
286:  int line(int x1, int y1, int x2, int y2,
287:           void (*action)(int,int,int),int mode)
288:  /***************************************************
289:
290:   Linien mit dem Integer-Bresenham-Verfahren
291:   zwischen den Punkte (x1,y1) und (x2,y2) zeichnen
292:
293:   <action> ist eine dreistellige Funktion, die
294:   bei jedem zu zeichnenden Punkt aufgerufen wird.
295:
296:   Bei mode = dick überlappen sich die Punkte an
297:   den Linienkanten, bei duenn logischerweise nicht
298:
299:   Ergebnis: Anzahl der gesetzten Punkte.
300:
301:  ***************************************************/
302:  {
303:    register int x,y;
304:
305:    int dx,dy,
306:        xstep,ystep,
307:        abw, points;
308:
```

→

```
309:    points = 1; /* Länge der Linie in Pixel */
310:    x = x1; y = y1;
311:    xstep = ystep =1;
312:
313:    dx = x2 - x1;
314:
315:    if (dx == 0) xstep = 0;
316:    else if (dx < 0) {
317:      dx = -dx;
318:      xstep = -xstep;
319:    }
320:
321:    dy = y2 - y1;
322:
323:    if (dy == 0) ystep = 0;
324:    else if (dy < 0) {
325:      dy = -dy;
326:      ystep = -ystep;
327:    }
328:
329:    abw = dx - dy;
330:
331:    while ((x != x2) || (y != y2)) {
332:      points++;              /* schon wieder'n Punkt */
333:      (*action)(x,y,1);  /* jetzt gibt's Äkschn   */
334:      if (abw >= 0) {
335:        x   += xstep;
336:        abw -= dy;
337:        if ((abw < 0) && mode) { /* mode == dick */
338:          points++;          /* Überlappende ..   */
339:          (*action)(x,y,1); /* ..Punkte setzen.  */
340:        }
341:      }
342:      if (abw < 0) {
343:        y   += ystep;
344:        abw += dx;
345:      }
346:    } /* end while */
347:    (*action)(x,y,1);
348:    return points; /* der Kandidat erhält 99 Gummi-
349:                      punkte */
350:  } /* end line */
351:
352:  void verform(int x0, int y0, /* Offsets */
353:               int x1, int y1, int x2, int y2,
354:               /* Startrechteck */
355:               int x3, int y3, int x4, int y4,
356:               /* Zielviereck */
357:               int x5, int y5, int x6, int y6)
358:  {
359:    int  pts_l1,pts_l2, /* Länge von Linie 1 u 2 */
360:         pos1,pos2,     /* Koordinatenzähler in
361:                           Punktlisten */
362:         laengst,     /* Länge der längsten Linie */
363:         oz;
364:    long step1,step2,/* Schrittweiten in den
365:                        Punktlisten   * 2^BITMOVE */
366:         pos1g,pos2g,ozg,
367:         steporig;
368:
369:    hide_mouse(); /* Schnell die Maus verstecken */
370:    z = 0; pts_l1 = line(x3,y3,x4,y4,list1,duenn);
371:    z = 0; pts_l2 = line(x6,y6,x5,y5,list2,duenn);
372:
373:    laengst  = max(pts_l1,pts_l2);
374:    step1    = ((long) pts_l1 << BITMOVE) /
375:               (long) laengst;
376:    step2    = ((long) pts_l2 << BITMOVE) /
377:               (long) laengst;
378:    steporig = ((long) (ABS(x2-x1)+1) << BITMOVE) /
379:               (long) laengst;
380:
381:    for (pos1 = pos2 = oz = 0,
382:         ozg = pos1g = pos2g = 0L;
383:         (pos1 < pts_l1) && (pos2 < pts_l2) ; ){
384:     makelist(ABS(xlist[0][pos1]-xlist[1][pos2])+
385:              ABS(ylist[0][pos1]-ylist[1][pos2])+1,
386:              oz+x1+x0,y0,y1,y2);
387:      z = 0;
388:      line(xlist[0][pos1],ylist[0][pos1],
389:           xlist[1][pos2],ylist[1][pos2],
390:           plot,dick);
391:      pos1g += step1;
392:         /* genauer Koordinatenzähler */
393:      pos2g += step2;
394:      ozg   += steporig;
395:      pos1 = (int) (pos1g >> BITMOVE);
396:         /* absoluter Koordinatenzähler */
397:      pos2 = (int) (pos2g >> BITMOVE);
398:      oz   = (int) (ozg >> BITMOVE);
399:      if (oz+x1+x0 > x2) {
400:        x0 = oz = 0;
401:        ozg = 0L;
402:      }
403:    }
404:    show_mouse(0); /* mit Käse fängt man Mäuse */
405:  } /* verform , war doch einfach, oder ? */
406:
407:  int get_pix(int x,int y,void *logbase)
408:  /*********************************************
409:
410:   Ermittelt die Farbe des Punktes an (x,y) in dem
411:   über logbase referenzierten logischen Bildschirm
412:   Die Funktion arbeitet auflösungsunabhängig.
413:
414:   Ergebnis: Punktfarbe
415:
416:  *********************************************/
417:  {
418:    int i,
419:        res=0,
420:        dist = (Planes-1);
421:    int *adr;
422:
423:    adr = (int *) ((char *) logbase + y *
424:           BytesZeile + ((x >> 4)<<PDBitmove));
425:
426:    for ( i = 0; i < Planes; i++){
427:      res <<= 1;
428:      res += (*((int *)( adr + dist)) &
429:             (1 << (15 - (x & 15))) ? 1 : 0);
430:      dist -= 1;
431:    }
432:    return res;
433:  } /* end get_pix */
434:
435:  void plot ( int x, int y, int col)
436:  /*********************************************
437:
438:   Setzt an (x,y) einen Punkt der Farbe colors[z],
439:   wenn
440:   - (x,y) innerhalb der Bildschirmgrenzen
441:   - transparent == TRUE und col != 0
442:   - transparent == FALSE (also deckend)
443:   - man ATARI-Fan ist.
444:
445:   <col> ist hier nur ein Dummywert, der durch
446:   ein Element aus colors[] ersetzt wird.
447:
448:   z und transparent sind global.
449:
450:  *********************************************/
451:
452:  {
453:    int farbe = colors[z++];
454:
455:    if (transparent) {
456:      if (farbe &&
457:          x>=0 && y>=0 && x<=MAXX && y<=MAXY)
458:       put_pixel(x,y,farbe); }
459:    else
460:      if (x>=0 && y>=0 && x<=MAXX && y<=MAXY)
461:        put_pixel(x,y,farbe);
462:  } /* end plot */
463:
464:  void list1(int x,int y, int dummy)
465:  {
466:    xlist[0][z] = x; ylist[0][z++] = y;
467:  } /* end (ganz schön listig, nich' wahr?) */
468:
469:  void list2(int x,int y, int dummy)
470:  {
471:    xlist[1][z] = x; ylist[1][z++] = y;
472:  }
473:  void makelist(int len,int x, int y0, int y1,
474:                int y2)
475:  /*********************************************
476:
477:   Abtasten einer Spalte (Koordinate x, zwischen y1
478:   und y2) im Quellbild und speichern der Farbwerte
479:   im <len>-grossen Feld colors, wobei nötigenfalls
480:   gestreckt / gestaucht wird.
481:
482:  *********************************************/
```

```
483:  {
484:    int y,z;
485:    long stepy,yg;
486:
487:    stepy = ((long)  (ABS(y2-y1)+1L) << BITMOVE) /
488:            (long) len;
489:
490:    for (z = y = 0, yg = 0L; z<=len; z++) {
491:      colors[z] = (char) get_pix(x,y+y0+y1,
492:                                 Original);
493:      yg += stepy;
494:      y = (int) (yg >> BITMOVE);
495:      if (y + y0 + y1 > y2) {y0 = y = 0;  yg = 0;}
496:    }
497:  } /* end makelist */
498:
499:  void    *ZweiterSchirm (void)
500:  /************************************************
501:
502:   Adresse eines zweiten Bildschirms ermitteln und
503:   einen PICLEN groPen Speicherbereich reservieren.
504:
505:   -> nix
506:   <- Adresse des neuen Bildschirms bzw.
507:      NULL bei Fehler
508:
509:  ***********************************************/
510:
511:  {
512:    void *base;
513:
514:    if ((base = malloc( PICLEN+256 )) == NULL)
515:      return NULL;
516:    else return ((void *) ((((size_t) base ) +
517:                              256) & 0xffffff00L));
518:  } /* end SonnenSchirm */
519:
520:  size_t BildLaden(void *base, char *Name)
521:  {
522:  /************************************************
523:
524:   Laden des Bildes <Name> an die Adresse <base>.
525:
526:    -> base  Basisadresse
527:    -> Name  Dateipfad und -name
528:    <- Fehler        0
529:    <- kein Fehler  Dateilänge
530:
531:  ***********************************************/
532:
533:    FILE *stream;
534:    size_t len;
535:
536:    if ((stream = fopen(Name,"rb")) == NULL )
537:      return 0;
538:    else {
539:      len = filelength(fileno(stream));
540:      if (fread(base,1,len,stream) == len) {
541:        fclose(stream);
542:        return len;}
543:      else {
544:        fclose(stream);
545:        return 0;
546:      }
547:    }
548:  } /* end BildGeschäft */
549:
550:  /**** Das war's. Viel Spass noch damit!      ****/
```

MSCROLL

MAUS AUSGELÖSTES CURSOR-SCROLLING

Peter Holzwarth

ZUR TEXTEINGABE UND -MODIFIKATION WERDEN AM ATARI ST TYPISCHERWEISE ZWEI EINGABEMEDIEN VERWENDET: DIE TASTATUR UND DIE MAUS. DABEI WERDEN MIT DER TASTATUR TEXTE EINGEGEBEN, UND MIT DER MAUS (ODER MIT DEM CURSOR-BLOCK) WIRD DER CURSOR AN DIE ENTSPRECHENDEN STELLEN BEWEGT BZW. DAS SICHTBARE TEXTFENSTER VERSCHOBEN. DAS TEXTFENSTER ZEILENWEISE MIT DEN CURSOR-TASTEN ZU VERSCHIEBEN, IST JEDOCH NICHT OPTIMAL, DENN DANACH MUSS ZUMEIST DER CURSOR AN EINE BESTIMMTE STELLE IM BILDSCHIRM BEWEGT WERDEN. DAS TEXTFENSTER ZEILENWEISE MIT DER MAUS ZU VERSCHIEBEN, IST 'BLIND' MEIST NICHT MÖGLICH, WEIL DIE MAUS DAZU (BEI GEM-EDITOREN) AUF EIN KLEINES PFEILFELD BEWEGT WERDEN MUSS.

Eine andere Lösung erscheint deshalb ergonomischer: Es wird vereinbart, daß der Cursor (und damit das sichtbare Textfenster) sich nach oben oder nach unten bewegen soll, sobald sich die Maus am betreffenden Bildrand befindet. So wird das Aufsuchen einer neuen Textstelle zu einem einzigen linearen Vorgang: Der Benutzer schiebt die Maus (fast ohne vom Text wegzusehen) an den oberen oder den unteren Bildrand und positioniert, wenn der gesuchte Textbereich sichtbar wird, den Cursor an der gewünschten Stelle.

Ein einziges Problem ergibt sich dabei leider: Wenn die Maus bei einem GEM-Editor in die oberste Bildzeile bewegt wird, klappt das sich an dieser Stelle befindende Menü herunter. Deshalb ist es hier notwendig, die Maus in die linke oder die rechte obere Ecke zu bewegen, wo sich in der Regel kein Menü befindet.

Ein Programm, das solch eine Scroll-Funktion implementiert, ist im Listing angegeben. Das Programm installiert eine neue VBL-Interrupt-Routine, in der die Mausposition abgefragt und bei entsprechender Stellung Tasten-Events im Tastaturpuffer abgelegt werden.

Die Arbeitsweise von M-SCROLL im Detail: M-SCROLL ist ein Accessory. Das ist notwendig, um es beim Systemstart gleich aktivieren zu können, denn Programme, die aus dem Autoordner gestartet werden, können keine VBL-Interrupt-Routine installieren und keine Line-A-Funktionen verwenden.

Die Initialisierungsroutine errichtet zuerst einen neuen Stack und besorgt sich dann die notwendigen Systeminformationen (Adresse der Line-A-Variablen, des Tastatur-Parameterblocks und die Auflösung) und installiert danach die neue VBL-Interrupt-Routine. Nachdem dies geschehen ist, wartet die Initialisierungsroutine auf einen Timer-Event mit Verzögerungszeit von -1. Das bedeutet, daß kein Rücksprung mehr erfolgt und das Betriebssystem mit dem Startvorgang fortfährt.

Der VBL-Interrupt (Vertical Blank Interrupt) wird je nach Auflösung und Voreinstellung beim Farbmonitor zwischen 50- und 70mal pro Sekunde aufgerufen. Dabei wird jedesmal die Routine *test* gestartet, die sich nun aus den Line-A-Variablen die aktuelle y-Position der Maus besorgt (negativer Offset 600). Wenn der Mauszeiger am oberen oder am unteren Bildrand steht, wird ein entsprechender Tasten-Event in den Tastaturpuffer eingetragen. Wenn der verwendete Editor jedoch eine GEM-Oberfläche besitzt, muß in der obersten Bildzeile der mittlere Bereich (in dem sich das Menü befindet) vom Scrolling ausgenommen werden, da sonst zwar das Menü ausgeführt wird, gleichzeitig aber auch Tasten-Events in das GEM-Message-System gelangen. Deshalb wird mit den Befehlen ab dem Symbol *.oben* getestet, ob sich die Maus in diesem Bereich befindet. Die Werte für *MitteLinks* und *MitteRechts* müssen ggf. dem verwendeten Editor angepaßt werden. Wird ein Editor ohne GEM-Oberfläche verwendet, sind die Werte für *MitteLinks* und *Mitte Rechts* gleichzusetzen.

Der Tastaturpuffer ist ein meist 128 Zeichen langer First In First Out-Puffer, in den die Daten wie auf einem Ring rundherum eingetragen werden. Dabei gibt es zwei Zeiger: *Tail* ist der aktuelle Schreib- und *Head* der aktuelle Lesezeiger. Wenn ein Zeichen in den Tastaturpuffer eingetragen wird, wird zuerst der Schreibzeiger erhöht (falls noch Platz im Puffer vorhanden ist) und dann am entsprechenden Offset das Zeichen eingetragen. Voll ist der Puffer, wenn der Schreibzeiger beim Erhöhen an den Lesezeiger anstößt. Leer ist der Puffer, wenn Schreib- und Lesezeiger gleich sind.

Auf diese Art und Weise werden also pro Sekunde bis zu 70 Zeichen in den Tastaturpuffer hineingefüllt. Da GEM-Programme nicht in der Lage sind, so viele Events pro Sekunde zu verarbeiten, muß, wenn der Mauszeiger ins Innere des Bildes zurückgeführt wird, eine entsprechende Nachbearbeitung des Tastaturpuffers stattfinden. Das besorgen die Befehle ab dem Symbol *.innen*: Wenn der Mauszeiger sich im Inneren des Bildes befindet, wird getestet, ob zuvor gescrollt wurde. In diesem Fall wird der Tastaturpuffer durch Gleichsetzen von Schreib- und Lesemarke gelöscht.

MSCROLL wurde mit dem GFA-Assembler entwickelt, ist aber auch auf jedem anderen Assembler ohne nennenswerte Anpassungen assemblierbar. Am Anfang des Quelltextes werden einige Macro-Definitionen includet, die zum bequemen Aufruf von Betriebssystemfunktionen verwendet werden. Besitzer des GFA-Assemblers können an dieser Stelle die Datei *ATARI.IS* includen und ersparen sich dadurch die Eingabe dieser Makros.

MSCROLL läuft in allen Auflösungen (Besitzer von Großbildschirmen müssen in den Befehlen ab dem Symbol *.resolution* die verwendete Auflösung eintragen), mit allen Betriebssystemversionen und mit allen Editoren, die die Maus sichtbar lassen. Getestet wurde das Accessory mit den Editoren Tempus, Wordplus und Altex (LPR-Modula2).



```
 1:  ;
 2:  ; für MScroll.IS benötigte Makro-Aufrufe für
       Betriebssystem-Funktionen
 3:  ; 30.6.1990 Peter Holzwarth
 4:  ; (c) MAXON Computer, 1991
 5:
 6:                 .MACRO Getrez
 7:                 move.w     #4,-(sp)
 8:                 trap       #14
 9:                 addq.w     #2,sp
10:                 .ENDM
11:
12:                 .MACRO Iorec devn
13:                 move.w     \1,-(sp)
14:                 move.w     #14,-(sp)
15:                 trap       #14
16:                 addq.w     #4,sp
17:                 .ENDM
18:
19:                 .MACRO Supexec code
20:                 move.l     \1,-(sp)
21:                 move.w     #38,-(sp)
22:                 trap       #14
23:                 addq.w     #6,sp
24:                 .ENDM
25:
26:  _vblqueue      = $456
27:                 .END
```

```
 1:  ; ***************************************
 2:  ; *                                     *
 3:  ; * Cursorscrolling durch Mausanschlag *
 4:  ; * 28.6.1990 Peter Holzwarth          *
 5:  ; * (c) MAXON Computer 1991            *
 6:  ; ***************************************
 7:
 8:                 .INCLUDE 'MSCRMAC.IS'
 9:
10:  GCURX          = -602     ; Offset der Maus-X-
                                 Position i.d.Line-A-Var.
11:  GCURY          = -600     ; Offset der Maus-Y-
                                 Position i.d.Line-A-Var.
12:  sc_up          = $48      ; Scan-Code Cursor hoch
13:  sc_do          = $50      ; Scan-Code Cursor runter
14:
15:  MitteLinks     = 12       ; Bereich i.d.Mitte d.
                                 Bildes, in dem die Maus
16:  MitteRechts    = 512      ; kein Aufwärtsscrolling
                                 auslösen soll
17:
18:                 .ABS 0
19:  buffer:        .DS.l 1    ; Anfang der Struktur des
                                 Parameterblocks
20:  Size:          .DS.w 1    ; für den Tastaturpuffer
21:  Head:          .DS.w 1
22:  Tail:          .DS.w 1
23:
```

```
24:                  .TEXT
25:  ; ***************************************
26:  ; Initialisierung
27:
28:  start:          lea        StackEnd,sp ; eigenen
                                            Stack einrichten
29:
30:  .appl_init:     moveq.l    #10,d0  ; wg. Accessory
31:                  moveq.l    #0,d1
32:                  moveq.l    #1,d2
33:                  moveq.l    #0,d3
34:                  bsr        aes
35:
36:  .LineA:         .DC.w $a000        ; Basisadresse der
                                          Line-A-Variablen
37:                  move.l     a0,LineAvars
38:
39:  .kbdvect:       Iorec      #1         ; Tastaturpuffer-
                                          Parameterblock holen
40:                  move.l     d0,KeyPar
41:
42:  .resolution:    Getrez     ;          ; Auflösung holen
43:                  move.w     #199,d1 ; 200 Y-Punkte in
                                          Lo- und MidRes
44:                  subq.w     #2,d0
45:                  bmi        .resok
46:                  move.w     #399,d1 ; 400 Y-Punkte in
                                          HiRes
47:  .resok:         move.w     d1,ymax
48:
49:  .vbl:           Supexec    #vblinit  ; s.u.
50:
51:  .evnt_timer:    move.l     #-1,intin ; Zeit=-1 (nie)
52:                  moveq.l    #24,d0
53:                  moveq.l    #2,d1
54:                  moveq.l    #0,d2
55:                  moveq.l    #0,d3
56:                  bsr        aes        ; Warten a.Ende
57:                  bra        .evnt_timer
58:
59:  vblinit:        movea.l    _vblqueue,a1
                                 ; VBL-Interrupt-Routine
                                 installieren
60:  .1:             tst.l      (a1)+      ; freien Platz
                                             suchen
61:                  bne        .1
62:                  move.l     #test,-(a1) ; und Routine
                                             eintragen
63:  .e:             rts
64:
65:  aes:            moveq.l    #0,d4    ; GEM-AES aufrufen
66:                  movem.w    d0-d4,contrl
67:                  move.l     #AESpar,d1
68:                  move.w     #200,d0
69:                  trap       #2
70:                  rts
71:
72:  ; ***************************************
73:  ; im VBL-Interrupt testen, ob Maus oben oder
       unten ist                                    →
```

```
 74:
 75:  test:          movea.l    LineAvars,a0
 76:                 move.w     GCURY(a0),d0
 77:                 beq        .oben
 78:                 cmp.w      ymax,d0
 79:                 beq        .unten
 80:
 81:  .innen:        tst.b      scrolling  ; Maus im
                                    Inneren des Bildschirms
 82:                 beq        .testend   ; hat zuvor
                                                Scrolling
                                            stattgefunden?
 83:                 movea.l    KeyPar,a0
 84:                 clr.l      Head(a0)     ; dann lösche
                                             Tastaturpuffer
 85:                 sf         scrolling  ; um Nachlaufen
                                             zu verhindern
 86:                 bra        .testend
 87:
 88:  .oben:         move.w     GCURX(a0),d0 ; Maus oben
 89:                 cmpi.w     #MitteLinks,d0 ; ist Maus
                                        im mittleren Teil des
 90:                 bcs        .obenOK       ; Bildes?
 91:                 cmpi.w     #MitteRechts,d0
 92:                 bcs        .innen        ; dann nicht
                                                scrollen
 93:  .obenOK:       st         scrolling
 94:                 moveq      #sc_up,d0
 95:                 bsr        send
 96:                 bra        .testend
 97:
 98:  .unten:        st         scrolling     ; Maus unten
 99:                 moveq      #sc_do,d0
100:                 bsr        send
101:
102:  .testend:      rts
103:
104:  send:          swap.w     d0 ; Taste in
                                  Tastaturpuffer eintragen
105:                 movea.l    KeyPar,a0    ; ($0048 0000
                                           oder $0050 0000)
106:                 movea.l    (a0),a1
107:                 move.w     Tail(a0),d1
108:                 addq.w     #4,d1
109:                 cmp.w      Size(a0),d1  ; wrapping
                                                notwendig?
110:                 bcs        .1
111:                 clr.w      d1
112:  .1:            cmp.w      Head(a0),d1 ; Puffer voll?
113:                 beq        .e
114:                 move.w     d1,Tail(a0) ; noch nicht,
                                        Schreibzeiger erhöhen
115:                 move.l     d0,0(a1,d1.w) ; und Taste
                                                 eintragen
116:  .e:            rts
117:
118:                 .DATA
119:  AESpar:        .DC.l contrl,globl,intin,intout,
                          addrin,addrout ; AES-
                                           Parameterblock
120:
121:                 .BSS
122:  contrl:        .DS.w 12   ; AES-Datenfelder
123:  globl:         .DS.w 15
124:  intin:         .DS.w 2
125:  intout:        .DS.w 1
126:  addrin:        .DS.l 1
127:  addrout:       .DS.l 1
128:                 .DS.b 4096   ; neuer Stack-Bereich
129:  StackEnd:
130:  LineAvars:     .DS.l 1    ; Basisadresse der Line-A-
                                  Variablen (neg. Offsets)
131:  KeyPar:        .DS.l 1    ; Tastaturpuffer-
                                  Parameterblock
132:  ymax:          .DS.w 1    ; max. Y-Koordinate der
                                  Maus
133:  scrolling:     .DS.b 2    ; -1: Scrolling wird
                                  gerade durch Maus
                                  ausgelöst
134:                 .END
```



Dem variablen Kontrollfeld auf der Spur Episode 2

Nachdem nun endlich die lange Zeit des Wartens auf den zweiten Teil vorüber ist (leider konnten wir sie nicht durch einen Kochkurs verkürzen), wenden wir uns wieder der Kontrollfeldprogrammierung zu. Nach der Behandlung der Grundlagen im ersten Teil kommen dieses Mal die restlichen, noch nicht besprochenen Kontrollfeldfunktionen an die Reihe.

Bevor wir uns jetzt mit den übrigen Funktionen befassen, noch ein Hinweis, den man unbedingt beachten sollte: Will man Funktionen benutzen, die interne Variablen verändern, die von anderen Programmen benutzt werden könnten, so sollte man diese unbedingt retten und nach Benutzung wiederherstellen. Dies betrifft insbesondere System- und Line-A-Variablen.

Um den Bitblt-Parameterblock zu retten, stellt das Kontrollfeld bereits eine entsprechende Funktion zur Verfügung (womit wir schon bei der ersten Funktion wären): *copy_bltparm()*. Sie sichert entweder die aktuellen Parameter in einen angegebenen Speicherbereich oder stellt sie wieder her.

Eine weitere nützliche Funktion ist *find_cookie()* (bereits im ersten Teil benutzt), denn durch den Cookie-Jar (siehe auch [1]) wird es möglich, z.B. den Rechnertyp festzustellen oder mit residenten Programmen zu kommunizieren. Mit anderen Worten: bei der Funktion *find_cookie()* handelt es sich mithin um eine der wichtigsten Funktionen für CPX-Module, denn wie in Teil 1 bereits erläutert, ist das CPX-Konzept insbesondere für die bequeme und einfache Konfigurierung residenter Programme gedacht.

Die meisten CPX-Module ermöglichen es, die eingestellten Parameter zu sichern. Das geschieht in der Regel dadurch, daß die Parameter direkt in die Programm-Datei eingetragen werden. Es wird also keine externe Parameter-Datei benötigt. Stattdessen wird vom Kontrollfeld die Funktion *write_config()* zur Verfügung gestellt. Zu beachten ist dabei, daß die zu sichernden Variablen unbedingt als allererste im Programm deklariert werden müssen, damit sie im Datensegment unmittelbar am Anfang stehen (siehe Listing „DISK.C" im ersten Teil). Also sollte man bei Header-Dateien, die vor diesen Variablen am Anfang eines Programms eingebunden werden, darauf achten, daß man in ihnen keine Variablen deklariert. Es empfiehlt sich auch, alle zu sichernden Variablen in einer Struktur zusammenzufassen, damit sich deren benötigte Größe bequem mit dem C-Operator *sizeof()* ermitteln läßt.

alert() stellt eine von zwölf bereits vordefinierten Alarmboxen im Kontrollfeldfenster zentriert dar. Allerdings sind die meisten Meldungen für eigene Programme nicht sinnvoll nutzbar.

Die Funktion *write_header()* schreibt einen neuen 512 Byte langen Header (wie in Teil 1 beschrieben), mit dessen Hilfe man beispielsweise ein neues Icon installieren kann.

Des weiteren gibt es noch die Funktion *multi()*, die ähnlich wie *evnt_multi()* funktioniert. Jedoch wartet *multi()* im Gegensatz zu *evnt_multi()* nicht, bis eine Message eintrifft, sondern kehrt nach der Installierung direkt wieder in das Modul zurück, so daß andere Programme nicht im Ablauf behindert werden (Stichwort: Multitasking!). Weitere Informationen zur Event-Behandlung folgen im letzten Teil dieser Trilogie.

Slider, unendliche Weiten...

Kommen wir nun zu den Schiebern, die auch im vorliegenden Programm eine wichtige Rolle spielen.

Unter Schiebern (neudeutsch: Slider) versteht man dabei die Objekte, die man zur Positionierung in den Randbereichen von Fenstern vorfindet und wohl jedem bekannt sein dürften. Sie zeigen zum einen durch ihre Größe an, wie groß der sichtbare Teil- im Verhältnis zum Gesamtbereich ist, und zum anderen zeigen sie die relative Position im Gesamtbereich an. Dazu sind einige Berechnungen erforderlich, die jedoch zum größten Teil das Kontrollfeld übernimmt.

Zur Nutzung von Schiebern muß beim Entwurf einer Dialogbox im Resource

Construction Set ein Slider folgendermaßen angelegt werden: Ein Slider besteht zum einen aus einem übergeordneten Objekt (meist eine G_BOX) als Hintergrund und zum anderen aus einem weiteren Objekt (meist ebenfalls eine G_BOX) innerhalb des Hintergrundes, das dann den eigentlichen Slider darstellt. Außerdem können rechts und links bzw. oben und unten Pfeile vom Typ G_BOXCHAR angegeben werden, wie ebenfalls aus Fenstern bekannt. Alle zum Slider gehörigen Objekte sollten sie dabei als TOUCHEXIT deklarieren.

Um die fertigen Slider zu manipulieren, stellt das Kontrollfeld folgende Funktionen zur Verfügung:

size_slider() ermöglicht die einfache Berechnung und Einstellung der Slider-Größe, *pos_vslider()* und *pos_hslider()* der Position für vertikale und horizontale Schieber. Bei beiden Funktionen muß das Neuzeichnen anschließend vom Modul selbst übernommen werden.

inc_slider() verschiebt den Slider jeweils um eine angegebene Einheit in vertikaler oder horizontaler Richtung. *move_vslider()* und *move_hslider()* werden benutzt, um einen vertikalen oder horizontalen Schieber beliebig innerhalb des übergeordneten Objekts verschieben zu können. Bei diesen Funktionen übernimmt das Kontrollfeld das Neuzeichnen des Schiebers.

Allen Slider-Funktionen außer *size_slider()* kann man darüber hinaus noch die Adresse einer Funktion übergeben, die während der Abarbeitung der Slider-Funktion ausgeführt wird (siehe auch Listing "BOOT.C" und dazugehörige Programmbeschreibung weiter unten).

Auf dem Holodeck

Betrachten wir einmal die beiden Abbildungen 1 und 2, die das fertige CPX-Modul zeigen. Wie der Name schon sagt, handelt es sich um einen Boot-Selektor, der wie die Boot-Selektoren arbeitet, die üblicherweise im AUTO-Ordner stehen. Aber natürlich muß nach der Auswahl der zu bootenden Programme und Accessories noch ein Reset ausgeführt werden. Wie aus Abbildung 2 hervorgeht, haben wir uns beim Design der Dialogbox sehr stark an der vergleichbaren Dialogbox für die CPX-Auswahl orientiert, um eine einheitliche Oberfläche zu gewährleisten.

"Captain's Log, Supplemental"

Kommen wir jetzt zur Programmbeschreibung, wobei wir auch dieses Mal wieder aufgrund der ausführlichen Kommentare

Übersicht über die Kontrollfeldfunktionen (Teil 2)

WORD alert(WORD number)

Anzeige einer im Kontrollfeldfenster zentrierten Alert-Box

Übergabeparameter:
number Nummer der Fehlermeldung
Rückgabe: Nummer des angeklickten Buttons
Vordefinierte Fehlermeldungen:

Nummer	*Text*	*Anzahl Buttons*
0	Voreinstellungen sichern?	2
1	Fehler bei Speicheranforderung!	1
2	Fehler beim Schreiben/Lesen von Dateien!	1
3	Datei nicht gefunden!	1
4	Zur Zeit können nicht mehr CPX geladen werden!	1
5	CPX neu laden?	2
6	CPX entfernen?	2
7	Residente CPX können nicht entfernt werden!	1
8	CPX-Konfiguration sichern?	2
9	Datei ist kein CPX. Schreiben abgebrochen!	1
10	Stereo-Sound nicht möglich!	1
11	Kontrollfeld abschalten?	2

VOID copy_bltparm(WORD dir, VOID *buffer)

Kopieren der BITBLT-Parameter

Übergabeparameter:
dir Kopierrichtung:
0: nach BITBLT
1: von BITBLT
buffer Adresse des Parameter-Buffers
Rückgabe: keine

VOID inc_slider(OBJECT *tree, WORD box_index, WORD slider_index, WORD button_index, WORD increment, WORD start, WORD total, WORD *slider_pos, WORD v_h_flag, VOID (*function)())

Automatisches Verschieben eines Sliders bei gedrückter Maustaste

Übergabeparameter:
tree Zeiger auf den Objektbaum (Dialogbox)
box_index Index des Objekts, das den Slider enthält
slider_index Index des Sliders
button_index Index eines Objekts, das selektiert werden soll, solange die Maustaste gedrückt ist
-1, falls kein Objekt selektiert werden soll
increment vorzeichenbehafteter Wert, der auf *slider_pos* aufaddiert wird
start kleinste Slider-Position im verschiebbaren Bereich
total größte Slider-Position im verschiebbaren Bereich
slider_pos Slider-Position nach Beendigung der Funktion;
start <= slider_pos <= total
v_h_flag 0 = vertikaler Slider
1 = horizontaler Slider
function Adresse einer Funktion, die bei Bearbeitung von *inc_slider()* ausgeführt wird
Rückgabe: evtl. Rückgabewert von *function()*, sonst keine

VOID move_hslider(OBJECT *tree, WORD box_index, WORD slider_index, WORD start, WORD total, WORD *slider_pos, VOID (*function)())

VOID move_vslider(OBJECT *tree, WORD box_index, WORD slider_index, WORD start, WORD total, WORD *slider_pos, VOID (*function)())

Verschieben eines horizontalen bzw. vertikalen Sliders in jede Richtung bei gedrückter Maustaste

Übergabeparameter und Rückgabe: wie bei *inc_slider()*

→

des Listings nur Schwerpunkte herausgreifen wollen.

Die Initialisierungsroutine *init()* funktioniert auf die gleiche Art und Weise wie bei DISK.CPX. Die einzige Ausnahme ist die Tatsache, daß hier keine Parameter mit Hilfe von Systemvariablen oder anderen systemabhängigen Einstellungen gesetzt oder gelesen werden müssen, da der Boot-Selektor seine Informationen direkt aus den aktivierten und deaktivierten Accessories bzw. Programmen bezieht.

main() stellt wie üblich das Hauptprogramm dar, in dem hier die augenblickliche Konfiguration eingelesen, die Slider initialisiert und dann die Dialogbox gezeichnet werden. Die Slider-Initialisierung, d.h. Einstellung der Größe und Position, erfolgt vor dem Zeichnen der Dialogbox - da wie bereits oben kurz erwähnt - diese beiden Slider-Funktionen das (Neu-)Zeichnen der Slider nicht mit übernehmen. Anschließend werden in einer Schleife alle angeklickten Objekte ausgewertet und die entsprechende Reaktion veranlaßt. Der Boot-Selektor funktioniert dabei auf genau die gleiche Art und Weise wie der CPX-Selektor im Kontrollfeld, d.h. sowohl die Dateinamen in den Fenstern, die Pfeile der Slider, als auch die Slider und deren Hintergrundobjekte selbst können angeklickt werden. Wird ein Objekt (Dateiname) in einem der Fenster selektiert, kann man die Datei daraufhin durch Auswahl von *aktivieren* bzw. *deaktivieren* entsprechend umbenennen.

get_traddr() funktioniert wie in DISK.-CPX und liefert die Adresse einer Dialogbox zurück.

read_config() liest automatisch alle aktiven bzw. inaktiven Programmnamen ein. Je nach übergebenem Programmtyp werden entweder die Accessories aus dem Hauptdirectory oder die Auto-Ordner-Programme aus dem AUTO-Ordner des zukünftigen Boot-Laufwerks eingelesen und in die entsprechenden Listen eingetragen. Läuft eine der beiden Listen dabei über, d.h. enthält sie mehr Einträge, als das Modul verwalten kann, erfolgt eine Warnmeldung mittels der Funktion *warning()*. Für die in den Listen enthaltenen Dateinamen funktioniert der Boot-Selektor jedoch uneingeschränkt auch weiterhin. Die Maximalzahl der verwaltbaren Namen läßt sich mit den Konstanten MAX_ACTIVE und MAX_INACTIVE einstellen; sie wurden so gewählt, daß höchstens ausgesprochene Software-Sammler in Bedrängnis kommen dürften.

unselect() schaltet einen selektierten Namen in einem der beiden Auswahlfenster ab und benutzt dazu die Funktion *change_object()*, die den entsprechenden Eintrag neu zeichnet.

VOID multi(WORD ev_flags, MOUSE_EVENT *mm1, MOUSE_EVENT *mm2, LONG timer)

Neue *evnt_multi()*-Routine

Übergabeparameter:

ev_flags — auszuwertende Ereignisse; entspricht *ev_mflags* aus *evnt_multi()*

mm1, mm2 — Werte für Mausereignisse; entspricht mehreren Einträgen in *evnt_multi()*. Wird kein Maus-Event benötigt, so ist NULL anzugeben. Die Struktur MOUSE_EVENT ist folgendermaßen definiert:

```
typedef struct
{
        WORD   flags;
        WORD   x,y,w,h;
} MOUSE_EVENT;
```

flags — 1, falls sich der Mauszeiger außerhalb des Rechtecks befindet und 0, falls innerhalb

x,y,w,h — Position und Größe des Rechtecks

timer — Wert für Timer-Ereignis; entspricht *ev_mthicount* und *ev_mtlocount* in *evnt_multi()*
Der Timer-Event ist aktiv, sobald ein von -1L verschiedener Wert angegeben wird.
Achtung: Damit durch das aktive Kontrollfeld andere Programme nicht vollständig gestoppt werden, erfolgt **automatisch** alle 30 Sekunden ein Timer-Event; dieser läßt sich nicht abschalten.

Rückgabe: keine

VOID pos_hslider(OBJECT *tree, WORD box_index, WORD slider_index, WORD slider_pos, WORD start, WORD total, VOID (*function)())

VOID pos_vslider(OBJECT *tree, WORD box_index, WORD slider_index, WORD slider_pos, WORD start, WORD total, VOID (*function)())

Positionieren eines horizontalen bzw. vertikalen Sliders im Gesamtbereich

Übergabeparameter und Rückgabe: wie bei *inc_slider()*; *slider_pos* ist hier jedoch als Wert zu übergeben

VOID size_slider(OBJECT *tree, WORD box_index, WORD slider_index, WORD total, WORD seen, WORD v_h_flag, WORD min_size)

Berechnung der Größe eines Sliders im Verhältnis zum Gesamtbereich

Übergabeparameter:

tree, box_index, slider_index, total, v_h_flag — wie bei *inc_slider()*

seen — Anzahl der sichtbaren Einträge, 0 <= *seen* <= *total*

min_size — minimale Slider-Größe in Pixeln

Rückgabe: keine

WORD write_config(VOID *params, LONG length)

Sichern der aktuellen Einstellung in die CPX-Datei

Übergabeparameter:

params — Adresse eines Speicherbereichs, der die Parameter enthält

length — Anzahl der zu schreibenden Bytes

WORD write_header(CPX_BLOCK *wrt_header)

Schreiben eines neuen CPX-Headers

Übergabeparameter:

wrt_header — Adresse einer Struktur, die alle für das Schreiben des CPX-Headers relevanten Informationen enthält; CPX_BLOCK ist in "XCONTROL.H" definiert. Der CPX-Header wird nur dann geschrieben, wenn *wrt_header->ok* TRUE ist.
Achtung: *wrt_header->header.magic*, *wrt_header->header.cpx_id* und *wrt_header->header.cpx_version* müssen mit den entsprechenden Werten des aktuellen CPX-Moduls übereinstimmen.

Rückgabe: TRUE falls Erfolg, sonst FALSE

redraw_object() ist die bereits in DISK.CPX besprochene und praktisch unveränderte Redraw-Routine, wie sie für das Kontrollfeld notwendig ist.

refresh() schaltet einen selektierten Dateinamen ab, und je nach Art des Namens (aktiv, inaktiv, leer) wird der Aktionsknopf entsprechend neu gezeichnet ("deaktivieren", "aktivieren" und leer). Die Funktion *get_entry()* liefert dazu den Index des selektierten Namens.

rename_file() nennt die ausgewählte Datei entsprechend um, um sie zu aktivieren bzw. deaktivieren. Dies geschieht wie üblich bei Boot-Selektoren durch Ersetzen des letzten Buchstabens des Dateinamens durch ein 'X'. Je nach Programmtyp (ACC oder PRG) werden die Zugriffspfade dabei automatisch entsprechend gewählt.

copy() entfernt die umbenannte Datei aus der aktiven Liste und trägt sie in der inaktiven Liste ein bzw. umgekehrt.

boot_device() ermittelt das nächste Boot-Laufwerk durch Auslesen der korrespondierenden Systemvariablen.

Die Adressen von *scroll_up()* und *scroll_down()* werden an einige Slider-Funktionen übergeben und dann bei deren Abarbeitung ausgeführt; sie scrollen dabei den Fensterinhalt eines der beiden Fenster nach oben oder unten. Beide Funktionen sind im Programm sowohl ohne Übergabeparameter als auch ohne Rückgabewert deklariert worden. Ein Rückgabewert ist hier nicht nötig, und die Übergabe von Parametern leider nicht möglich, da den Slider-Funktionen nur die Adresse einer auszuführenden Routine übergeben werden kann. Deshalb wurde die (in unserem Programm benötigte) Parameterübergabe über eine globale Variable (*scroll_param*) durchgeführt. Ein eventueller Rückgabewert wird von der ausführenden Slider-Funktion zurückgegeben.

into_resource() gibt eine Liste von Dateinamen in einem der beiden Auswahlfenster aus.

pos_vslider() positioniert einen vertikalen Slider in einem der Fenster unter Berücksichtigung der Ober- und Untergrenze neu, wenn die Hintergrundbox, in der sich der Slider befindet, angeklickt wurde. Zu beachten ist bei allen Slider-Funktionen folgende Eigenheit, die jedesmal eine Überprüfung der Slider-Grenzen notwendig macht: Es darf in dem Übergabeparameter total nur die Obergrenze des **verschiebbaren** Bereichs angegeben werden, d.h. der sichtbare Bereich (repräsentiert durch die Slider-Größe) muß eigenhändig vom Gesamtbereich abgezogen werden. Außerdem ist zu beachten, daß sich bei vertikalen Slidern die Position Null am unteren Ende befindet, was leider im Normalfall ebenfalls ein Umrechnen der sichtbaren Einträge erforderlich macht. (Wer verwaltet seine Tabellen oder Texte schon von unten nach oben?)

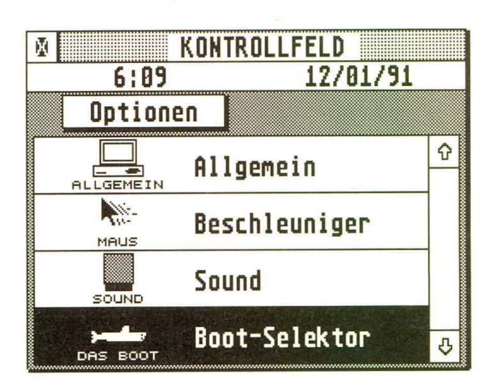


Abb. 1: Das installierte Modul BOOT.CPX

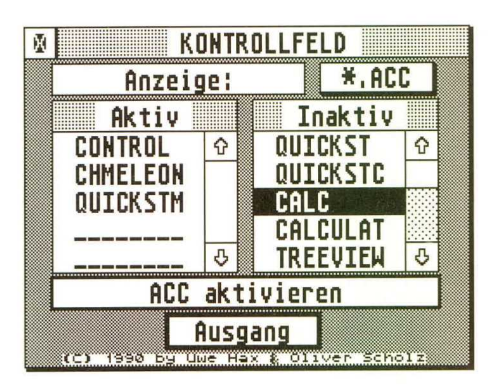


Abb. 2: Die Dialogbox des Moduls BOOT.CPX

Die einzelnen Werte für die Slider-Positionierung sollen an folgendem einfachen Beispiel kurz demonstriert werden. Stellen wir uns vor, wir möchten eine Liste mit zwölf Einträgen anzeigen, die der Einfachheit halber durchnumeriert sind; der darzustellende Bereich sei fünf Zeilen groß. Das ganze sieht dann in etwa wie in Abbildung 3 aus.

Die Slider-Position ist in diesem Fall 1 und der verschiebbare Bereich umfaßt 12-5=7 Einheiten (*Text1* bis *Text6* und *Text12*), also müssen die Übergabeparameter *start* und *total* beispielsweise 0 und 6 lauten.

Aber jetzt noch die restlichen Programmroutinen: Die Funktion *move_vslider()* reagiert auf das Verschieben des Sliders mit der Maus und zeichnet den Fensterinhalt entsprechend der neuen Slider-Position.

init_bd() und *init_slider()* führen einige Initialisierungen aus, die der Übersicht halber aus *main()* ausgelagert wurden.

wind_center() arbeitet wie in DISK.CPX und zentriert eine Dialogbox im Kontrollfeldfenster.

Schließlich bleibt noch die Routine *pulldown()*, die ähnlich wie im ersten Teil das Pull-Down-Menü generiert und bei Änderung entsprechend des ausgewählten Eintrags die Konfiguration einliest und darstellt.

Alle im Listing BOOT.C benutzten und im ersten Teil noch nicht aufgeführten Kontrollfeldfunktionen sind im Kasten noch einmal übersichtlich und in alphabetischer Reihenfolge mit Beschreibung aller Übergabeparameter aufgelistet. Zu beachten ist wieder, daß es sich bei allen Funktionen tatsächlich um Zeiger auf Funktionen handelt und alle als *cdecl* deklariert sind; die in der Tabelle verwendete Schreibweise ist jedoch übersichtlicher.

Wesley über die andere Schulter geschaut

Um nun aus den abgedruckten Listings ein lauffähiges CPX-Modul zu erhalten, muß man (fast) genauso wie beim letzten Mal vorgehen, d.h. aus den Listings XCONTROL.H (man kann es unverändert aus dem ersten Teil übernehmen), BOOT.H, BOOT.RSH, und BOOT.C muß mittels der Projektdatei BOOT.PRJ ein (nicht ausführbares) BOOT.PRG erstellt werden. Anschließend muß man dann noch mit Hilfe von BBUILD.PRG das CPX-Modul BOOT.CPX erstellen. Wer das Listing DBUILD.C aus dem ersten Teil bereits

abgetippt hat, muß dort nur alle Initialisierungsparameter ändern, um daraus die Datei BBUILD.C zu erhalten, woraus sich durch Compilieren und Linken mit der Header-Datei XCONTROL.H BBUILD.-PRG erstellen läßt.

Das alles klingt recht aufwendig (keine Angst: ist es auch!), aber zur vermutlich großen Beruhigung aller Programmierer können wir in der nächsten Folge u.a. ein "CPX-Construction-Kit" anbieten, das dem CPX-Entwickler diese umständliche Prozedur abnimmt. Man darf also gespannt sein... Nichtsdestotrotz ist unsere Dokumentation an dieser Stelle bereits so komplett, daß jeder ohne Probleme eigene CPX-Module entwickeln können müßte. Und keine Panik, wenn nicht alles auf Anhieb funktioniert. Das Motto lautet: "Don't worry, be trekkie!"

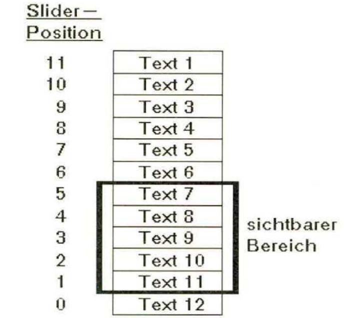


Abb. 3: Beispiel für die Berechnung der Slider-Werte

Uwe Hax & Oliver Scholz

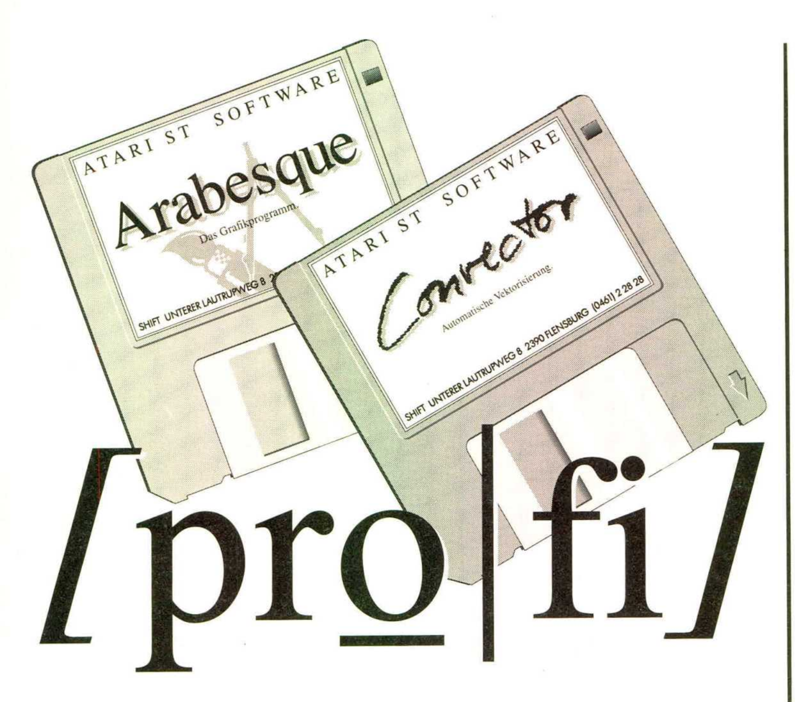
[1] Rolf Kotzian:
STee-Gebäck - Das Cookie-Jar-Prinzip
ST-Computer 12/90, S. 151 ff.


```
  1: /**********************************************/
  2: /* Datei: BBUILD.C                            */
  3: /* ——————————————————— */
  4: /* Modul: BOOT.CPX              Version 1.00 */
  5: /* (C) 1990 by MAXON Computer                */
  6: /* Autoren: Uwe Hax & Oliver Scholz          */
  7: /* verwendeter Compiler: Turbo-C 2.0         */
  8: /**********************************************/
  9:
 10:
 11: /* die üblichen Header-Dateien ———— */
 12:
 13: #include <portab.h>
 14: #include <tos.h>
 15: #include <string.h>
 16: #include <stdlib.h>
 17: #include <aes.h>
 18:
 19:
 20: struct foobar   /* ist normalerweise in   */
 21: {               /* "*.rsh" definiert und  */
 22:   WORD dummy;   /* wird in "xcontrol.h"   */
 23:   WORD *image;  /* benötigt               */
 24: };
 25:
 26: #include "xcontrol.h"
 27:
 28:
 29: /* Definitionen zur besseren Lesbarkeit —— */
 30:
 31: #define SOURCE        "BOOT.PRG"
 32: #define DESTINATION   "BOOT.CPX"
 33:
 34: #define TRUE          1
 35: #define FALSE         0
 36:
 37:
 38: /* globale Variablen ——————— */
 39:
 40: /* Header-Definiton */
 41: CPX_HEADER header;
 42:
 43: /* Image-Daten */
 44: LONG data[24]={  0x00000000L, 0x00000000L,
 45:                  0x00000000L, 0x00000000L,
 46:                  0x00000000L, 0x00000000L,
 47:                  0x00000000L, 0x00000000L,
 48:                  0x00000000L, 0x00000000L,
 49:                  0x00001c00L, 0x00003c00L,
 50:                  0x00003c00L, 0xc0007c00L,
 51:                  0xe3ffffffL, 0x7fffffbL,
 52:                  0xfffffffeL, 0x7fffffaL,
 53:                  0xe07ffffcL, 0xc0000000L,
 54:                  0x00000000L, 0x00000000L,
 55:                  0x00000000L, 0x00000000L
 56:              };
 57:
 58:
 59: /* Prototypen für Turbo-C ——————— */
 60:
 61: VOID main(VOID);
 62: VOID abort_main(VOID *buffer,WORD fd);
 63:
 64:
 65: VOID main(VOID)
 66: {
 67:   VOID *buffer;
 68:   DTA *dta=Fgetdta();
 69:   WORD fd;
 70:   WORD i;
 71:
 72:   /* Kennung für *.CPX-Datei */
 73:   header.magic=100;
 74:
 75:   /* Bitvektor: Flags für Lade-Modus */
 76:   header.flags.set_only=FALSE;
 77:   header.flags.boot_init=FALSE;
 78:   header.flags.resident=FALSE;
 79:
 80:   /* Kurzkennung */
 81:   strncpy(header.cpx_id,"BOOT",4);
 82:
 83:   /* Versionsnummer */
 84:   header.cpx_version=0x100;
 85:
 86:   /* Icon-Name */
 87:   strcpy(header.icon_name,"DAS BOOT");
 88:
 89:   /* Image-Daten */
 90:   for (i=0; i<24; i++)
 91:     header.icon_data[i]=data[i];
 92:
 93:   /* Icon: Farbe 4, kein Buchstabe */
 94:   header.icon_info=0x4000;
 95:
 96:   /* Programmname */
 97:   strcpy(header.cpx_name,"Boot-Selektor");
 98:
 99:   /* Farben */
100:   header.obj_state=0x1280;
101:
102:
103:   /* Header und Programm zusammenbauen */
104:
105:   if (Fsfirst(SOURCE,0)<0)
106:     abort_main(0L,-1);
107:
108:   if ((buffer=Malloc(dta->d_length))<0)
109:     abort_main(0L,-1);
110:
111:   if ((fd=Fopen(SOURCE,0))<0)
112:     abort_main(buffer,-1);
113:
114:   if (Fread(fd,dta->d_length,buffer)<0)
115:     abort_main(buffer,fd);
116:   Fclose(fd);
117:
118:   if ((fd=Fcreate(DESTINATION,0))<0)
119:     abort_main(buffer,-1);
120:
121:   if (Fwrite(fd,512L,&header)!=512L)
122:     abort_main(buffer,fd);
123:
124:   if (Fwrite(fd,dta->d_length,buffer)!=dta-
        >d_length)
125:     abort_main(buffer,fd);
126:
127:   Mfree(buffer);
128:   Fclose(fd);
129:   exit(0);
130: }
131:
132:
133: VOID abort_main(VOID *buffer,WORD fd)
134: {
135:   if (buffer)
136:     Mfree(buffer);
137:   if (fd>=0)
138:     Fclose(fd);
139:   form_alert(1,"[3][CPX-Datei konnte nicht|\
140: erzeugt werden!][ Abbruch ]");
141:   exit(1);
142: }
```

Listing 1: BBUILD.C

```
 1: ;**********************************************
 2: ;* Datei: BOOT.PRJ                            *
 3: ;* ——————————————— *
 4: ;* Modul: BOOT.CPX              Version 1.00 *
 5: ;* (C) 1990 by MAXON Computer                *
 6: ;* Autoren: Uwe Hax & Oliver Schulz          *
 7: ;* Projektdatei für Turbo-C 2.0              *
 8: ;**********************************************
 9:
10: boot.prg
11: =
12: boot.c
13: tcstdlib.lib
14: tcgemlib.lib
15: tctoslib.lib
```

Listing 2: BOOT.PRJ



Arabesque ist durch die Tool-Box-Serie noch professioneller geworden. Ihr erstes Modul: ***Convector***, das Programm zur automatischen Vektorisierung.

Es ist durch spezielle Schnittstellen besonders für die Zusammenarbeit mit Arabesque ausgelegt und wandelt beliebige Grafiken oder Bildschirmausschnitte in Vektorgrafiken um, die dann (unter anderem) mit Arabesque nachbearbeitet werden können.

Auch von Arabesque gibt's Neuigkeiten. ***Arabesque Professional*** ist lieferbar. Die neue Pro-Version erweitert Arabesque um Bezier-Polygone und unterstützt sowohl das GEM/3 als auch das Calamus®- Format für Vektorgrafiken.

Arabesque und Convector sind die professionellen Lösungen für Atari ST und TT. Die richtige Software für Ihre Gestaltungsarbeiten. Zu einem fairen Preis.

SHIFT
UNTERER LAUTRUPWEG 8
2390 FLENSBURG
☎ (0461) 2 28 28 FAX 1 70 50

SCHWEIZ: EDV-DIENSTLEISTUNGEN
ERLENSTRASSE 73
8805 RICHTERSWIL
☎ (01) 784 89 47

ÖSTERREICH: AMV-BÜROMASCHINEN
MARIAHILFERSTRASSE 77-79
1060 WIEN
☎ (0222) 586 30 30

NIEDERLANDE: MOPRO
POSTBUS 2293
3500 GG UTRECHT
☎ (030) 31 62 47

SHIFT. Sachen gibt's...









Convector Automatische Vektorisierung.

Arabesque Die Grafikprogramme.

THEMADAT Assoziative Datenbank.

CyPress Die Textverarbeitung.

dBMAN

in Deutsch

für ST/TT DM 599-

CHAMÄLEON

Wissenschaftliche Karteiverwaltung
für ST/TT DM 149,-

SCHRÖTTLE - SHELL V

Unix-Bourne Shell

für ST/TT DM 99,-

Modula 2

DM 349,-

1ST MATHLAB

DM 249,-

ANALYZE

Lotus 1-2-3 kompatible Tabellenkalkulation für ST/TT DM 299,-

Redakteur

ST-Textverarbeitung DM 69,-

SPECTRE GCR

Macintosh-Emulator o. ROM's
DM 599,-

SYNTEX

OCR-Texterkennung DM 199,-

600 DPI

Laserkit DM 349,-

COMPUTER MAI
GmbH & Co. Software KG
Metzstr. 19, 8000 München 80
Tel.: 089/4480691

```
 1:  /************************************************/
 2:  /* Datei: BOOT.H                                */
 3:  /* ---------------------- */
 4:  /* Modul: BOOT.CPX                 Version 1.00 */
 5:  /* (C) 1990 by MAXON Computer                   */
 6:  /* Autoren: Uwe Hax & Oliver Scholz             */
 7:  /* Vom RCS aus Resource-Datei erstellte         */
 8:  /* Include-Datei                                */
 9:  /************************************************/
10:
11:
12:  #define BOOT 0           /* TREE */
13:  #define TYP 2            /* OBJECT in TREE #0 */
14:  #define ACTWIND 5        /* OBJECT in TREE #0 */
15:  #define ACTIVE1 6        /* OBJECT in TREE #0 */
16:  #define ACTIVE3 7        /* OBJECT in TREE #0 */
17:  #define ACTIVE2 9        /* OBJECT in TREE #0 */
18:  #define ACTIVE4 8        /* OBJECT in TREE #0 */
19:  #define ACTIVE5 10       /* OBJECT in TREE #0 */
20:  #define ACT_UP 11        /* OBJECT in TREE #0 */
21:  #define INACTIV1 13      /* OBJECT in TREE #0 */
22:  #define INACWIND 12      /* OBJECT in TREE #0 */
23:  #define INACTIV2 14      /* OBJECT in TREE #0 */
24:  #define INACTIV3 15      /* OBJECT in TREE #0 */
25:  #define INACTIV5 17      /* OBJECT in TREE #0 */
26:  #define INACTIV4 16      /* OBJECT in TREE #0 */
27:  #define INACT_UP 18      /* OBJECT in TREE #0 */
28:  #define PARENT1 19       /* OBJECT in TREE #0 */
29:  #define PARENT2 21       /* OBJECT in TREE #0 */
30:  #define SLIDER1 20       /* OBJECT in TREE #0 */
31:  #define SLIDER2 22       /* OBJECT in TREE #0 */
32:  #define ACT_DOWN 23      /* OBJECT in TREE #0 */
33:  #define ACTION 25        /* OBJECT in TREE #0 */
34:  #define INACT_DO 24      /* OBJECT in TREE #0 */
35:  #define QUIT 26          /* OBJECT in TREE #0 */
36:  #define ERROR 1          /* TREE */
37:  #define MIST 6           /* OBJECT in TREE #1 */
```

Listing 3: BOOT.H

```
 1:  /************************************************/
 2:  /* Datei: BOOT.RSH                              */
 3:  /* ---------------------- */
 4:  /* Modul: BOOT.CPX                 Version 1.00 */
 5:  /* (C) 1990 by MAXON Computer                   */
 6:  /* Autoren: Uwe Hax & Oliver Scholz             */
 7:  /* Vom RCS aus Resource-Datei erstellte         */
 8:  /* Include-Datei                                */
 9:  /************************************************/
10:
11:
12:  #define NUM_FRSTR 0
13:  #define NUM_FRIMG 0
14:  #define NUM_OBS 35
15:  #define NUM_TREE 2
16:
17:
18:  BYTE *rs_strings[] =
19:  {
20:    "Anzeige:","","","*.ACC"," Aktiv ","","",
21:    " Inaktiv ","","","__________","","","__________",
22:    "","","__________","","","__________","","",
23:    "__________","","","__________","","","__________",
24:    "","","__________","","","__________","","",
25:    "__________","","","ACC deaktivieren","","",
26:    "Ausgang",
27:    "(C) 1990 by Uwe Hax & Oliver Scholz",
28:    "","","Die maximale Anzahl der",
29:    "verwaltbaren Dateien","wurde überschritten!",
30:    "Sie sollten Ihre Festplatte",
31:    "mal wieder aufräumen!","Mist!"
32:  };
33:
34:  LONG rs_frstr[] =
35:  {
36:    0
37:  };
38:
39:  BITBLK rs_bitblk[] =
40:  {
41:    0
42:  };
43:
44:  LONG rs_frimg[] =
45:  {
46:    0
47:  };
48:
49:  ICONBLK rs_iconblk[] =
50:  {
51:    0
52:  };
53:
54:  TEDINFO rs_tedinfo[] =
55:  {
56:    (char *)0L, (char *)1L, (char *)2L, 3, 6, 2,
57:    0x1180, 0x0, 255, 9,1,
58:    (char *)4L, (char *)5L, (char *)6L, 3, 6, 2,
59:    0x11A1, 0x0, 255, 8,1,
60:    (char *)7L, (char *)8L, (char *)9L, 3, 6, 2,
61:    0x11A1, 0x0, 255, 10,1,
62:    (char *)10L, (char *)11L, (char *)12L, 3, 6, 2,
63:    0x1180, 0x0, 255, 9,1,
64:    (char *)13L, (char *)14L, (char *)15L, 3, 6, 2,
65:    0x1180, 0x0, 255, 9,1,
66:    (char *)16L, (char *)17L, (char *)18L, 3, 6, 2,
67:    0x1180, 0x0, 255, 9,1,
68:    (char *)19L, (char *)20L, (char *)21L, 3, 6, 2,
69:    0x1180, 0x0, 255, 9,1,
70:    (char *)22L, (char *)23L, (char *)24L, 3, 6, 2,
71:    0x1180, 0x0, 255, 9,1,
72:    (char *)25L, (char *)26L, (char *)27L, 3, 6, 2,
73:    0x1180, 0x0, 255, 9,1,
74:    (char *)28L, (char *)29L, (char *)30L, 3, 6, 2,
75:    0x1180, 0x0, 255, 9,1,
76:    (char *)31L, (char *)32L, (char *)33L, 3, 6, 2,
77:    0x1180, 0x0, 255, 9,1,
78:    (char *)34L, (char *)35L, (char *)36L, 3, 6, 2,
79:    0x1180, 0x0, 255, 9,1,
80:    (char *)37L, (char *)38L, (char *)39L, 3, 6, 2,
81:    0x1180, 0x0, 255, 9,1,
82:    (char *)40L, (char *)41L, (char *)42L, 3, 6, 2,
83:    0x1180, 0x0, 254, 17,1,
84:    (char *)44L, (char *)45L, (char *)46L, 5, 6, 0,
85:    0x1180, 0x0, 255, 36,1
86:  };
87:
88:  OBJECT rs_object[] =
89:  {
90:    -1, 1, 27, G_BOX, NONE, NORMAL, 0xFF1141L,
91:    0,0, 32,11,
92:    2, -1, -1, G_BOXTEXT, NONE, NORMAL, 0x0L,
93:    2,768, 274,257,
94:    3, -1, -1, G_BUTTON, 0x41, SHADOWED, 0x3L,
95:    789,768, 8,1,
96:    4, -1, -1, G_BOXTEXT, NONE, NORMAL, 0x1L,
97:    2,2049, 525,257,
98:    5, -1, -1, G_BOXTEXT, NONE, NORMAL, 0x2L,
99:    1040,2049, 525,257,
100:   11, 6, 10, G_BOX, NONE, NORMAL, 0xFF1100L,
101:   2,2562, 1802,5,
102:   7, -1, -1, G_TEXT, 0x51, NORMAL, 0x3L,
103:   0,0, 1802,1,
104:   8, -1, -1, G_TEXT, 0x51, NORMAL, 0x4L,
105:   0,2, 1802,1,
106:   9, -1, -1, G_TEXT, 0x51, NORMAL, 0x5L,
107:   0,3, 1802,1,
108:   10, -1, -1, G_TEXT, 0x51, NORMAL, 0x6L,
109:   0,1, 1802,1,
110:   5, -1, -1, G_TEXT, 0x51, NORMAL, 0x7L,
111:   0,4, 1802,1,
112:   12, -1, -1, G_BOXCHAR, TOUCHEXIT, NORMAL,
113:   0x1FF1100L, 13,2562, 514,1,
114:   18, 13, 17, G_BOX, NONE, NORMAL, 0xFF1100L,
115:   1040,2562, 1802,5,
116:   14, -1, -1, G_TEXT, 0x51, NORMAL, 0x8L,
117:   0,0, 1802,1,
118:   15, -1, -1, G_TEXT, 0x51, NORMAL, 0x9L,
119:   0,1, 1802,1,
120:   16, -1, -1, G_TEXT, 0x51, NORMAL, 0xAL,
121:   0,2, 1802,1,
122:   17, -1, -1, G_TEXT, 0x51, NORMAL, 0xBL,
123:   0,3, 1802,1,
124:   12, -1, -1, G_TEXT, 0x51, NORMAL, 0xCL,
125:   0,4, 1802,1,
126:   19, -1, -1, G_BOXCHAR, TOUCHEXIT, NORMAL,
127:   0x1FF1100L, 1051,2562, 514,1,
128:   21, 20, 20, G_BOX, TOUCHEXIT, NORMAL,
129:   0xFF1111L, 13,2819, 514,3586,
130:   19, -1, -1, G_BOX, TOUCHEXIT, NORMAL,
131:   0xFF1100L, 0,0, 514,258,
132:   23, 22, 22, G_BOX, TOUCHEXIT, NORMAL,
133:   0xFF1111L, 1051,2819, 514,3586,
134:   21, -1, -1, G_BOX, TOUCHEXIT, NORMAL,
135:   0xFF1100L, 0,0, 514,258,
136:   24, -1, -1, G_BOXCHAR, TOUCHEXIT, NORMAL,
137:   0x2FF1100L, 13,2566, 514,1,
138:   25, -1, -1, G_BOXCHAR, TOUCHEXIT, NORMAL,
139:   0x2FF1100L, 1051,2566, 514,1,
140:   26, -1, -1, G_BOXTEXT, 0x41, NORMAL, 0xDL,
141:   2,3591, 1563,1,
142:   27, -1, -1, G_BUTTON, 0x7, NORMAL, 0x2BL,
143:   778,1033, 1290,1,
144:   0, -1, -1, G_TEXT, LASTOB, NORMAL, 0xEL,
145:   1282,2314, 1306,1536,
146:   -1, 1, 6, G_BOX, NONE, OUTLINED, 0x21100L,
147:   0,0, 797,3848,
148:   2, -1, -1, G_STRING, NONE, NORMAL, 0x2FL,
149:   1538,3840, 23,1,
150:   3, -1, -1, G_STRING, NONE, NORMAL, 0x30L,
151:   516,3841, 21,1,
152:   4, -1, -1, G_STRING, NONE, NORMAL, 0x31L,
153:   1028,3842, 20,1,
154:   5, -1, -1, G_STRING, NONE, NORMAL, 0x32L,
155:   257,2308, 27,1,
156:   6, -1, -1, G_STRING, NONE, NORMAL, 0x33L,
157:   771,2309, 21,1,
158:   0, -1, -1, G_BUTTON, 0x27, NORMAL, 0x34L,
159:   778,1031, 8,1
160: };
161:
162:
163: LONG rs_trindex[] =
164: {
165:   0L,27L
166: };
167:
168: struct foobar
169: {
170:   WORD     dummy;
171:   WORD     *image;
172: } rs_imdope[] = {
173:                 0
174:               };
```

Listing 4: BOOT.RSH

→

```
  1: /**************************************************/
  2: /* Datei: BOOT.C                                  */
  3: /* ————————————————— */
  4: /* Modul: BOOT.CPX                 Version 1.00 */
  5: /* (C) 1990 by MAXON Computer                     */
  6: /* Autoren: Uwe Hax & Oliver Scholz               */
  7: /* verwendeter Compiler: Turbo-C 2.0              */
  8: /**************************************************/
  9:
 10:
 11: /* die üblichen Header-Dateien ———— */
 12:
 13: #include <portab.h>
 14: #include <aes.h>
 15: #include <tos.h>
 16: #include <stdlib.h>
 17: #include <string.h>
 18: #include <vdi.h>
 19:
 20:
 21: /* keine Variablen zu sichern */
 22:
 23: #include "boot.rsh"
 24: #include "boot.h"
 25: #include "xcontrol.h"
 26:
 27:
 28: /* Definitionen zur besseren Lesbarkeit —— */
 29:
 30: #define TRUE           1      /* Standard-Def. */
 31: #define FALSE          0
 32: #define EOS            '\0'
 33: #define NIL            0L
 34:
 35: #define MESSAGE        -1     /* Message-Event */
 36:
 37: #define MAX_ACTIVE     50   /* maximale Anzahl */
 38: #define MAX_INACTIVE   50   /* von Einträgen   */
 39: #define VISIBLE        5    /* sichtbare Eintr.*/
 40: #define NAME_LENGTH    8+1  /* Länge Eintrag   */
 41:
 42: #define MIN_SIZE       6    /* Slider-Größe    */
 43:
 44: #define ACTIVE         TRUE   /* Flags */
 45: #define INACTIVE       FALSE
 46: #define HORIZONTAL     1
 47: #define VERTICAL       0
 48: #define ACC            1
 49: #define PRG            0
 50:
 51: #define _bootdev       0x447 /* Systemvariable */
 52:
 53:
 54: /* globale Variablen ——————— */
 55:
 56: typedef struct
 57: {
 58:   WORD max_num;  /* Anzahl maximaler Einträge */
 59:   WORD num;      /* Anzahl Einträge */
 60:   WORD begin;    /* erster sichtbarer Eintrag */
 61:   WORD type;     /* ACC oder PRG */
 62:   WORD selected; /* Nr. selektierter Eintrag */
 63:   char (*buffer)[NAME_LENGTH];  /* Buffer für
 64:                                  Programmnamen */
 65: } BD;  /* Buffer-Deskriptor */
 66:
 67: BD active, inactive;  /* Deskriptoren für aktive
 68:                          und inaktive Programme */
 69:
 70: CPX_PARAMS *params; /* vom Kontrollfeld über-
 71:                        gebener Zeiger auf die
 72:                        Kontrollfeld-Funktionen */
 73:
 74: OBJECT *boot;        /* Zeiger auf Dialogboxen */
 75: OBJECT *error;
 76:
 77: WORD scroll_param;   /* Übergabe-Parameter für's
 78:                         Scrollen */
 79:
 80:              /* Indices der Dialogbox-Einträge */
 81: WORD act[VISIBLE]={ ACTIVE1, ACTIVE2, ACTIVE3,
 82:                     ACTIVE4, ACTIVE5 };
 83: WORD inact[VISIBLE]= { INACTIV1, INACTIV2,
 84:                    INACTIV3, INACTIV4, INACTIV5 };
 85:
 86: char empty[]="";        /* Strings für Dialogbox */
 87: char underlined[]="________";
 88: char acc[]="*.ACC";
 89: char prg[]="*.PRG";
 90: char activate[]="ACC aktivieren";
 91: char deactivate[]="ACC deaktivieren";
 92:
 93:
 94: /* Prototypen für Turbo-C —————— */
 95:
 96: char boot_device(VOID);
 97: VOID change_object(OBJECT *tree, WORD object,
 98:                    WORD state);
 99: VOID copy(BD *dest, BD *source, WORD index);
100: WORD get_entry(WORD object);
101: OBJECT *get_traddr(WORD tree_index);
102: CPX_INFO * cdecl init(CPX_PARAMS *params);
103: VOID init_bd(char (*act_buff)[NAME_LENGTH],
104:              char (*inact_buff)[NAME_LENGTH]);
105: VOID init_slider(WORD *slider1_pos,
106:                  WORD *slider2_pos, WORD draw);
107: WORD cdecl main(GRECT *curr_wind);
108: VOID into_resource(BD *bd, WORD draw);
109: VOID move_vslider(OBJECT *tree, WORD parent,
110:                   WORD slider, WORD *slider_pos,
111:                   BD *bd);
112: VOID pos_vslider(OBJECT *tree, WORD parent,
113:                  WORD slider, WORD *slider_pos,
114:                  BD *bd);
115: VOID pulldown(WORD *slider1_pos,
116:               WORD *slider2_pos);
117: VOID read_config(char *type, BD *descriptor);
118: VOID redraw_object(OBJECT *tree, WORD object);
119: VOID refresh(WORD active_flag, WORD object);
120: VOID scroll_down(VOID);
121: VOID scroll_up(VOID);
122: VOID unselect(WORD *array);
123: VOID rename_file(VOID);
124: VOID warning(VOID);
125: VOID wind_center(OBJECT *tree, WORD *x, WORD *y,
126:                  WORD *w, WORD *h);
127:
128:
129: /* Funktionen ——————— */
130:
131: /**************************************************/
132: /* Initialisierung des Moduls:                    */
133: /* Übergabeparameter: Zeiger auf die zur          */
134: /*    Verfügung stehenden Funktionen              */
135: /* 1. Aufruf bei Laden des Headers                */
136: /*    (par->booting == TRUE)                      */
137: /*    Rückgabe: 0 bei Set-Only, 1 sonst           */
138: /* 2. Aufruf bei Laden des eigentlichen           */
139: /*    Programms (par->booting == FALSE)           */
140: /*    Rückgabe: Adresse der CPX_INFO-Struktur     */
141: /**************************************************/
142:
143: CPX_INFO * cdecl init(CPX_PARAMS *par)
144: {
145:   static CPX_INFO info={ main,0L,0L,0L,0L,
146:                          0L,0L,0L,0L,0L };
147:
148:   if (par->booting)  /* bei Laden des Headers */
149:     return((CPX_INFO *)1L);
150:   else /* Aufruf bei Laden des Programms */
151:   {     /* => Löschen aller globalen Variablen! */
152:     params=par;  /* Zeiger retten! */
153:
154:     /* Resource relozieren */
155:     if (!params->rsc_init)
156:       (*params->do_resource)(NUM_OBS,NUM_FRSTR,
157:        NUM_FRIMG,NUM_TREE,rs_object,rs_tedinfo,
158:        rs_strings,rs_iconblk,rs_bitblk,rs_frstr,
159:        rs_frimg,rs_trindex,rs_imdope);
160:
161:     /* globale Variablen initialisieren */
162:     boot=get_traddr(BOOT);
163:     error=get_traddr(ERROR);
164:     strncpy(activate,"ACC",3);
165:     strncpy(deactivate,"ACC",3);
166:
167:     /* Dialogbox initialisieren */
168:     boot[ACTION].ob_flags=NONE;
169:     boot[ACTION].ob_spec.tedinfo->te_ptext=empty;
170:     strcpy(boot[TYP].ob_spec.free_string,acc);
171:
172:     /* Adresse der CPX_INFO-Struktur zurück */
173:     return(&info);
174:   }
175: }
176:
177:
178: /**************************************************/
179: /* Aufruf nach Doppelclick auf das Icon im        */
180: /* Auswahlfenster: Zeichnen der Dialogbox,        */
181: /* Behandlung der Buttons                         */
182: /* Übergabeparameter: Koordinaten des Fenster-    */
183: /*                    arbeitsbereichs             */
184: /* Rückgabe: FALSE, wenn der Dialog mittels       */
185: /*           do_form() abgearbeitet wird,         */
186: /*           TRUE, falls eigene Event-Routinen    */
187: /*           benutzt werden sollen                */
188: /**************************************************/
189:
190: WORD cdecl main(GRECT *curr_wind)
191: {
192:   WORD msg_buff[8];
193:   WORD button;
194:   WORD abort_flag=FALSE;
195:   char active_buff[MAX_ACTIVE][NAME_LENGTH];
196:   char inactive_buff[MAX_INACTIVE][NAME_LENGTH];
197:   WORD slider1_pos;
198:   WORD slider2_pos;
199:   WORD increment;
200:   VOID (*function)();
201:   WORD max;
202:
203:   /* Koordinaten der Dialogbox setzen */
204:   boot[ROOT].ob_x=curr_wind->g_x;
205:   boot[ROOT].ob_y=curr_wind->g_y;
206:
207:   /* Buffer-Deskriptoren initialiseren */
208:   init_bd(active_buff,inactive_buff);
209:
210:   /* Konfiguration einlesen... */
211:   read_config("*.ACC",&active);
212:   read_config("*.ACX",&inactive);
213:
214:   /* ...und in die Dialogbox eintragen */
215:   into_resource(&active,FALSE);
216:   into_resource(&inactive,FALSE);
217:
218:   /* Slider-Größe und Position initialisieren */
219:   init_slider(&slider1_pos,&slider2_pos,FALSE);
220:
221:   /* und Dialogbox zeichnen */
222:   objc_draw(boot,ROOT,MAX_DEPTH,boot[ROOT].ob_x,
223:             boot[ROOT].ob_y,boot[ROOT].ob_width,
224:             boot[ROOT].ob_height);
225:
226:   /* Dialogbox abarbeiten, bis ein Exit-Objekt      →
```

```
227:        angeklickt wurde */
228:     do
229:     {
230:       /* neuer form_do()-Aufruf */
231:       button=(*params->do_form)(boot,0,msg_buff);
232:
233:       /* Doppelklick ausmaskieren */
234:       if (button>=0)
235:         button &= 0x7fff;
236:
237:       /* Slider-Variablen setzen */
238:       increment=1;
239:       function=scroll_up;
240:
241:       /* angeklicktes Objekt auswerten */
242:       switch (button)
243:       {
244:         /* Name im "Aktiv"-Fenster angeklickt */
245:         case ACTIVE1:
246:         case ACTIVE2:
247:         case ACTIVE3:
248:         case ACTIVE4:
249:         case ACTIVE5:
250:           refresh(ACTIVE,button);
251:           break;
252:
253:         /* Name im "Inaktiv"-Fenster angeklickt */
254:         case INACTIV1:
255:         case INACTIV2:
256:         case INACTIV3:
257:         case INACTIV4:
258:         case INACTIV5:
259:           refresh(INACTIVE,button);
260:           break;
261:
262:         /* Datei aktivieren/deaktivieren */
263:         case ACTION:
264:           rename_file();
265:           into_resource(&active,TRUE);
266:           into_resource(&inactive,TRUE);
267:           init_slider(&slider1_pos,&slider2_pos,
268:                       TRUE);
269:           refresh(ACTIVE,button);
270:           break;
271:
272:         /* "Pfeil" im "Aktiv"-Fenster angeklickt */
273:         case ACT_DOWN:
274:           increment=-1;
275:           function=scroll_down;
276:
277:         case ACT_UP:
278:           scroll_param=ACTIVE;
279:           max=((max=active.num-VISIBLE)<0) ? 0 :
280:                max;
281:           (*params->inc_slider)(boot,PARENT1,
282:             SLIDER1,button,increment,0,max,
283:             &slider1_pos,VERTICAL,function);
284:           break;
285:
286:         /* "Pfeil" im "Inaktiv"-Fenster */
287:         case INACT_DO:
288:           increment=-1;
289:           function=scroll_down;
290:
291:         case INACT_UP:
292:           scroll_param=INACTIVE;
293:           max=((max=inactive.num-VISIBLE)<0) ? 0 :
294:                max;
295:           (*params->inc_slider)(boot,PARENT2,
296:             SLIDER2,button,increment,0,max,
297:             &slider2_pos,VERTICAL,function);
298:           break;
299:
300:         /* Slider angeklickt */
301:         case SLIDER1:
302:           move_vslider(boot,PARENT1,SLIDER1,
303:                        &slider1_pos,&active);
304:           break;
305:
306:         case SLIDER2:
307:           move_vslider(boot,PARENT2,SLIDER2,
308:                        &slider2_pos,&inactive);
309:           break;
310:
311:         /* Slider-Hintergrund angeklickt */
312:         case PARENT1:
313:           pos_vslider(boot,PARENT1,SLIDER1,
314:                       &slider1_pos,&active);
315:           break;
316:
317:         case PARENT2:
318:           pos_vslider(boot,PARENT2,SLIDER2,
319:                       &slider2_pos,&inactive);
320:           break;
321:
322:         /* "Anzeige-Typ" angeklickt */
323:         case TYP:
324:           pulldown(&slider1_pos,&slider2_pos);
325:           break;
326:
327:         /* Dialogbox verlassen */
328:         case QUIT:
329:           abort_flag=TRUE;
330:           break;
331:
332:         /* Nachricht eingetroffen */
333:         case MESSAGE:
334:           switch (msg_buff[0])
335:           {
336:             case WM_REDRAW:   /* nicht notwendig */
337:               break;
338:
339:             case WM_CLOSED: /* nichts zu sichern */
340:
341:             case AC_CLOSE:
342:               abort_flag=TRUE;
343:               break;
344:           }
345:           break;
346:       }
347:     }
348:     while (!abort_flag);
349:     boot[button].ob_state &= ~SELECTED;
350:     return(FALSE);
351: }
352:
353:
354: /**********************************************/
355: /* Liefert Adresse einer Dialogbox             */
356: /* (neue rsrc_gaddr()-Routine)                 */
357: /* Übergabeparamter: Baum-Index                */
358: /* Rückgabe: Zeiger auf Dialogbox              */
359: /**********************************************/
360:
361: OBJECT *get_traddr(WORD tree_index)
362: {
363:   WORD i,j;
364:
365:   for (i=0,j=0; i<=tree_index; i++)
366:     while (rs_object[j++].ob_next!=-1);
367:
368:   return(&rs_object[--j]);
369: }
370:
371:
372: /**********************************************/
373: /* Einlesen der aktivierten und deaktivierten  */
374: /* Programme                                   */
375: /* Übergabeparameter: Programmtyp, Adresse     */
376: /*                    des Buffer-Deskriptors   */
377: /* Rückgabe: keine                             */
378: /**********************************************/
379:
380: VOID read_config(char *type, BD *descriptor)
381: {
382:   DTA *dta=Fgetdta();
383:   WORD i=0;
384:   WORD j,k;
385:   char path[20];
386:   path[0]=boot_device();
387:
388:   /* Pfad je nach Typ aufbauen */
389:   if (!strcmp(type,"*.ACC") ||
390:       !strcmp(type,"*.ACX"))
391:     strcpy(&path[1],":\\");
392:   else
393:     strcpy(&path[1],":\\AUTO\\");
394:   strcat(path,type);
395:
396:   /* alle Namen dieses Typs einlesen */
397:   if (!Fsfirst(path,0))
398:     do
399:     {
400:       j=0;
401:       while (dta->d_fname[j]!='.')
402:         descriptor->buffer[i][j]=
403:                               dta->d_fname[j++];
404:       for (k=j; k<NAME_LENGTH-1; k++)
405:         descriptor->buffer[i][k]=' ';
406:       descriptor->buffer[i++][k]=EOS;
407:     }
408:     while (!Fsnext() && (i<descriptor->max_num));
409:   descriptor->num=i;
410:   if (descriptor->num==descriptor->max_num)
411:     warning();
412: }
413:
414:
415: /**********************************************/
416: /* Im Fenster selektierten Namen deselektieren */
417: /* Übergabeparameter: Adresse eines Feldes,    */
418: /*                    das die Indices der Fen- */
419: /*                    stereinträge enthält     */
420: /* Rückgabe: keine                             */
421: /**********************************************/
422:
423: VOID unselect(WORD *array)
424: {
425:   WORD i;
426:
427:   for (i=0; i<VISIBLE; i++)
428:     if (boot[array[i]].ob_state & SELECTED)
429:     {
430:       change_object(boot,array[i],NORMAL);
431:       if (array==act)
432:         active.selected=-1;
433:       else
434:         inactive.selected=-1;
435:       break;
436:     }
437: }
438:
439:
440: /**********************************************/
441: /* Objekt mit neuem Status zeichnen            */
442: /* Übergabeparameter: Zeiger auf Dialogbox,    */
443: /*                    Objekt-Index, neuer      */
444: /*                    Status                   */
445: /**********************************************/
446:
447: VOID change_object(OBJECT *tree, WORD object,
448:                    WORD state)
449: {
450:   WORD x,y;
451:
452:   objc_offset(tree,object,&x,&y);
```

→

OBERLAND SOFT & HARDWARE
Dietmar Schramm Promberg 6
8122 Penzberg Tel.: 08856 / 7287

SPEICHERAUFRÜSTUNG

AUF	260/520	1040 ST	MEGA 1	MEGA 2	1040STE
1MB	188,-	****	****	****	****
2MB	****	398,-	398,-	****	310,-
2.5MB	573,-	573,-	573,-	****	****
3MB	****	586,-	586,-	****	****
4MB	778,-	778,-	778,-	573,-	600,-

Calamus-Fonts
Mato-Vektor-Zeichensätze im Schriftenpaket. Pro Paket bis zu 26 Fonts. Je Paket
Nur 99,-
Calamus ist eingetragenes Warenzeichen der Firma DMC

Über 200 PD-Zeichensätze für Signum und Script. Für 9, 24 und Laserdrucker. Jeder Font nur 1,-DM

PD-Disketten 3,5" ab 4,50 DM
Alle PDs aus ST-Computer, PD-Pool, viele Pakete. Farbbänder für alle gängigen Drucker.

Kein Ladenverkauf! Selbstabholung n.v.m. kostenlose Liste anfordern



RAT & TAT

ERSATZTEIL-SERVICE

Adam-Opel-Straße 7–9 • W-6000 Frankfurt/Main 61

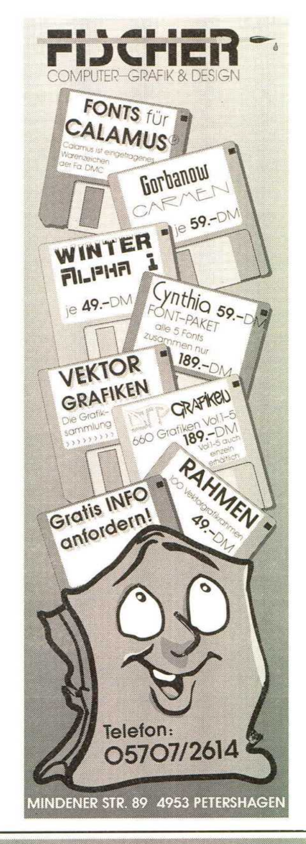
STÄNDIG ÜBER 200 VERSCHIEDENE
ORIGINAL-ATARI-ERSATZTEILE AM LAGER
ATARI 800 XL, 800 XE, 600 XL, 130 XE
Floppy 1050, 520/1040 ST

Netzteil für 800XL	49,95 DM	Bestell-Nr.	47708-1303
Netzteil für 1050	42,95 DM	Bestell-Nr.	47708-1053
IC Custom Glue	115,00 DM	Bestell-Nr.	47708-0008
IC Video Shifter	140,00 DM	Bestell-Nr.	47708-0010
IC Custom Glue	115,00 DM	Bestell-Nr.	47708-0011

FARBBÄNDER IN REICHHALTIGER AUSWAHL
Unverbindlich empfohlene Verkaufspreise

Erkundigen auch Sie sich nach unserem Lieferprogramm. Händleranfragen erwünscht. Preisänderungen vorbehalten. Versand per Nachnahme.

069/404-8769 • FAX 069/42 52 88 • BTX *41101#



ST-Fibu

Das Programmpaket
für Ihr Büro auf
ATARI ST/TT

FIBU

- ST-Fibu- komplette Finanzbuchhaltung mit Offener Postenverwaltung	ab DM 348.00
- ST-Fibu-Mini-Version	ab DM 148.00
- GMa-Text-Textverarbeitung mit Mahnwesen/Serienbrief *	ab DM 149.00
- ST-Fakt-Fakturierung *	ab DM 199.00
- ST-Inven-Inventarverwaltung	ab DM 79.00
Neu: ST-Giro- Abwicklung des Zahlungsverkehrs *	ab DM 99.00

* Programm mit Schnittstelle zur ST-Fibu

Demoversionen mit Handbuch DM 60.00 (wird beim Kauf verrechnet)
Kostenlose Info anfordern!



Gerd Matthäus
Betriebswirt

Bergstr. 18 - 6050 Offenbach
Tel. 069 / 89 83 45 - Fax 89 84 21



Wirkt nachhaltig gegen chronischen Ärger mit der Buchhaltung.

Wirkstoffe: 100.000e wohldosierter Bytes

Anwendungsgebiete:
Problemlose Einnahme-Überschuß-Rechnung (fibuMAN e + m) und Finanzbuchhaltung nach dem neuesten Bilanzrichtliniengesetz (fibuMAN f + m)

Nebenwirkungen:
exzellente Verträglichkeit mit:
fibuSTAT - graphische Betriebsanalyse
faktuMAN - modulares Business-System

Gegenanzeigen:
Verschwendungssucht, akute Aversionen gegen einfache und übersichtliche Buchhaltung

fibuMAN Programme gibt es schon ab DM 398,-
* unverbindliche Preisempfehlung Atari ST, Preise für fibuMAN MS-DOS® und Apple Macintosh® auf Anfrage

Testsieger in DATA WELT, 6/89
4 MS-DOS® Buchführungsprogramme im Prüfstand; davon 3 mit 8.23, 8.25, 8.65 Punkten (max. 10) fibuMAN mit der höchsten Punktzahl des Tests 9.35

fibuMAN begeistert Anwender wie Fachpresse! Nachzulesen in: ct 4/88, DATA WELT 3/88, 6/88, 5/89, 6/89, ST-COMPUTER 12/87, 12/88, ST-MAGAZIN 4/88, 10/88, ATARI SPECIAL 1/89, ATARI MAGAZIN 8/88, ST-PRAXIS 5/89, ST-VISION 3/89, PC-PLUS 5/89

NEU
1ST fibuMAN
Die Einsteiger-Buchführung
DM 148,-*

novoPLAN SOFTWARE GMBH
Hardtstraße 21 · 4784 Rüthen 3
Tel. (02952) 8080 + (0161) 2215791
Telefax (02952) 3236

Senden Sie mir für fibuMAN ○ INFO ○ Demo mit Handbuch
Ich arbeite mit dem System ○ MS-DOS ○ ATARI ○ Macintosh
Mein Name:
in Firma:
Straße/Nr.:
PLZ/Ort:
Demo mit Handbuch DM 65,-

Hardware ✧ Software

CameoST

Daß auch professionelle Software nicht zwangsläufig teuer sein muß, beweisenwir mit unserem CameoST, einem wahren Multitalent.

CameoST, das Musikarchiv

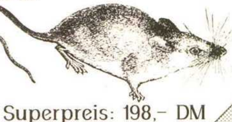
CameoST ist eine Datenbank für CDs, LPs und MCs. CameoST verfügt über flexible Suchfunktionen, die nicht nur alle Beethoveneinspielungen oder alle Versionen von 'Just the two of us', sondern auch alle verliehenen LPs in Sekundenschnelle ermittelt. Fest ins CameoST-Konzept eingebunden ist eine Adresskartei, in der Sie allen gespeicherten Personen auch beliebige Eigenschaftskennzeichen zuweisen können. Eine weitere Funktion ist der eingebaute Notizblock. **Einen ausführlichen Testbericht entnehmen Sie bitte der Zeitschrift 'PD-Journal' 11/90.**

Superpreis: 59,- DM

Cordless Mouse

Die innovative Infrarottechnik, die das lästige Mauskabel unnötig macht und ein ergonomisches Design stellen das Original weit in den Schatten. Dabei ist die Installation so einfach wie bei der alten Maus und Kompatibilitätsprobleme treten auch nicht auf.

Näheres erfahren Sie in dem Testbericht der 'TOS 10/90' bzw. 'PD-Journals' 11/90.



Superpreis: 198,- DM

Sonderaktion

Wir bieten Ihnen Vector-Fonts aus eigener Herstellung für Calamus✧. Damit Sie eine reichhaltige Auswahl an Schriften zu einem wirklich günstigen Preis erhalten, haben wir ein Schriftenpaket für Sie zusammengestellt. Dieses Paket enthält 200 Schriften. Diese Anzeige wurde übrigens mit dem Vektorfont 'Serif', der auch in dem Paket enthalten ist, gesetzt und belichtet.

200 Schriften nur: 249,-

✧ Calamus ist eingetragenes Warenzeichen der Firma DMC

Showtime^Pro

Dieses graphische Präsentation- und Informatinsprogramm reizt die Möglichkeiten Ihres ST voll aus. Allein 40 absolut flimmerfreie, saubere Umblendalgorithmen, GOTO, REPEAT....UNTIL- Strukturen, Multitasking-Sound, Vernetzung vieler Ataris mittels MIDI und die perfekte Benutzerführung machen dieses Programm zum Novum. Innerhalb kürzester Zeit erstellen Sie perfekte Diashows oder nutzen , durch die beliebigen Sprungmöglichkeiten auf Tastendruck, das Programm zum Aufbau eines Informations- bzw. Lernssystem. Demo 10,- DM

Interesse?

Für weitere Informationen stehen wir Ihnen gerne zur Verfügung.

Einführungspreis: 99,- DM

Speicher

Speichererweiterung für AtariST Computer von 512 KB auf 1 MB oder von 512KB, 1MB auf 2,5/4 MB. Die Speichererweiterung ist voll steckbar und für alle Computertypen (außer MegaST), deren MMU gesockelt ist, geeignet. Die große Aufrüstung kann in zwei Stufen erfolgen (2,5/4MB).

1 MB	198,- DM
2 MB	449,- DM
4 MB	698,- DM
Atari STE auf 2 MB	298,- DM
Atari STE auf 4 MB	549,- DM

Auf Wunsch übernehmen wir auch den Einbau

Einbau incl. Porto 50,- DM

Scanner

Bildverarbeitung für jedermann?

Ja. Denn jetzt gibt es Repro Studio junior. Dazu gehört nicht nur ein Logitech Handscanner mit 100-400 dpi, 32 Graustufen, 3 Raster, 105 mm Rasterbreite, sondern auch das professielle Bildverarbeitungsprogramm Repro Studio ST junior. Das alles erhalten Sie zu einem Preis, den Sie bisher nur für einen Scanner bezahlt hätten.

Scanner incl. Reprostudio 598,- DM

NEU !! Paket incl. 256 Graustufen Scanner 1.198,- DM

Hardware

Overscan	120,- DM
Testbericht PD-Journal 1/91	
LOGIMOUSE	89,- DM
That's a Mouse	94,- DM
BMC Track Ball	198,- DM
Best Modem 2400L	298,- DM
Best Modem 2448LF (Fax)	398,- DM
ProScreen TT 19Zoll	1.998,- DM
3,5 " Laufwerk	239,- DM
5.25 Laufwerk (40/80)	289,- DM

Datenbank Textverarbeitung

That's Write	298,- DM
I.D.A.	358,- DM
Signum!2	a.A.,- DM
Script 2	278,- DM
Phoenix	a.A.,- DM
Piccolo	99,- DM
STAD	169,- DM
That's Pixel	139,- DM
Tempus Word	569,- DM

Zubehör

10 Disketten 2DD	14,90 DM
50 Disketten	69,00 DM
10 Disk Maxell MF2-DD	24,90 DM
10 Disk Maxell MF2-HD	39,90 DM
Monitor Switchbox	49,90 DM
Auto Switchbox	59,90 DM
Druckerkabel 2m	29,90 DM
Scart-Kabel	29,90 DM
Verlängerungskabel Festplatte	49,90 DM
Farbband NEC P6	17,90 DM
Farbband Epson LQ 500,800,850	14,90 DM
Farbband Star LC10	11,90 DM

Software

Syntex	199,-	Reprok	549,-
X-Boot	69,-	Büroorganisation	
NVDI	94,-	Steinberg 12	99,-
Saldo	79,-	Cubase	949,-
Interlink	79,-	Printing Press professional	94,-
Neodesk 3.0	89,-		
Diskus	129,-	Fast Filemover	59,-
Tempus	109,-		

BTX-Software

MultiTerm pro an Modem	158,-
MultiTerm pro an DBT03	236,-

NEU NEU NEU

AT-Speed C16 DM 589,-

16MHz, Steckplatz für Co-Prozessor, inkl DOS-Betriebssystem !!

Avant-Vektor DM 648,-

die erste Echtzeitvektorisierung und vollautomatisch optimierende Vektorisierung.

Demo 10,- DM

Hersteller- und Händleranfragen erwünscht.

Wir suchen noch gute Soft- und Hardwareprodukte zum Vertrieb oder zur Vermarktung.

Versandkosten

Der Versand erfolgt per Nachnahme oder Vorauskasse. Natürlich können Sie alle Produkte auch telefonisch per Nachnahme bestellen. Die Versandkosten betragen 3,00 DM bei Scheck- oder Barzahlung und 6,00 DM bei Nachnahme.



02164/7898
0211/429876
BTX: Wohlfahrtstätter

Werner Wohlfahrtstätter ■ Irenenstraße 76c ■ 4000 Düsseldorf 30 ■ Tel.: 0211/429876

```
453:    objc_change(tree,object,0,x,y,
454:                tree[object].ob_width,
455:                tree[object].ob_height,state,TRUE);
456:  }
457:
458:
459:
460:  /*************************************************/
461:  /* Neuzeichnen eines Objekts mit Hilfe der vom */
462:  /* Kontrollfeld gelieferten Rechteck-Liste.    */
463:  /* Übergabeparameter: Zeiger auf Objektbaum,   */
464:  /*                    Objekt-Index             */
465:  /* Rückgabe: keine                             */
466:  /*************************************************/
467:
468:  VOID redraw_object(OBJECT *tree, WORD object)
469:  {
470:    GRECT *clip_ptr,clip,xywh;
471:
472:    /* absolute Objekt-Koordinaten berechnen */
473:    objc_offset(tree,object,&xywh.g_x,&xywh.g_y);
474:    xywh.g_w=tree[object].ob_width;
475:    xywh.g_h=tree[object].ob_height;
476:
477:    /* erstes Rechteck holen */
478:    clip_ptr=(*params->rci_first)(&xywh);
479:
480:    /* solange noch Rechtecke da sind */
481:    while (clip_ptr)
482:    {
483:      /* clip_ptr: Zeiger auf lokale Variable!! */
484:      clip=*clip_ptr;  /* deshalb kopieren */
485:
486:      /* Objekt neu zeichnen */
487:      objc_draw(tree,object,MAX_DEPTH,clip.g_x,
488:                clip.g_y,clip.g_w,clip.g_h);
489:
490:      /* nächstes Rechteck holen */
491:      clip_ptr=(*params->rci_next)();
492:    }
493:  }
494:
495:
496:  /*************************************************/
497:  /* Selektierten Eintrag im Fenster deselektie- */
498:  /* ren und je nach angeklicktem Eintrag den    */
499:  /* Aktions-Knopf neu zeichnen                  */
500:  /* Übergabeparameter: Fensterkennung,          */
501:  /*                    angeklicktes Objekt      */
502:  /* Rückgabe: keine                             */
503:  /*************************************************/
504:
505:  VOID refresh(WORD active_flag, WORD object)
506:  {
507:    /* Ausgabetext bestimmen */
508:    char *status=((active_flag==ACTIVE) ?
509:                  deactivate : activate);
510:
511:    /* selektierten Eintrag im anderen Fenster
512:       deselektieren */
513:    unselect((active_flag==ACTIVE) ? inact : act);
514:
515:    /* kein gültiger Eintrag angeklickt */
516:    if ((boot[object].ob_spec.tedinfo->te_ptext==
517:          underlined) || (object==ACTION))
518:    {
519:      /* selektierten Eintrag im aktuellen Fenster
520:         deselektieren */
521:      unselect((active_flag==ACTIVE) ?
522:               act : inact);
523:
524:      /* Aktions-Knopf abschalten */
525:      boot[ACTION].ob_spec.tedinfo->te_ptext=empty;
526:      boot[ACTION].ob_state &= ~SELECTED;
527:      boot[ACTION].ob_flags=NONE;
528:      redraw_object(boot,ACTION);
529:    }
530:    else
531:    {
532:      /* gültiger Eintrag angeklickt */
533:      if (boot[ACTION].ob_spec.tedinfo->te_ptext!=
534:          status)
535:      {
536:        /* Aktions-Knopf einschalten */
537:        boot[ACTION].ob_spec.tedinfo->te_ptext=
538:          status;
539:        boot[ACTION].ob_flags=SELECTABLE|TOUCHEXIT;
540:        redraw_object(boot,ACTION);
541:      }
542:
543:      /* Index des selektierten Eintrags merken */
544:      if (status==activate)
545:        inactive.selected=inactive.begin+
546:                          get_entry(object);
547:      else
548:        active.selected=active.begin+
549:                        get_entry(object);
550:    }
551:  }
552:
553:
554:  /*************************************************/
555:  /* Index des angeklickten Eintrags ermitteln   */
556:  /* Übergabeparameter: angeklicktes Objekt      */
557:  /* Rückgabe: gesuchter Index                   */
558:  /*************************************************/
559:
560:  WORD get_entry(WORD object)
561:  {
562:    WORD i;
563:
564:    for (i=0; i<VISIBLE; i++)
565:      if ((object==act[i]) || (object==inact[i]))
566:        break;
567:    return(i);
568:  }
569:
570:
571:  /*************************************************/
572:  /* Ausgewählte Datei umbenennen                */
573:  /* Übergabeparameter: keine                    */
574:  /* Rückgabe: keine                             */
575:  /*************************************************/
576:
577:  VOID rename_file(VOID)
578:  {
579:    WORD i;
580:    WORD *array;
581:    WORD index;
582:    WORD begin;
583:    char path1[30],path2[30];
584:    char (*buffer)[NAME_LENGTH];
585:    char flag=boot[ACTION].ob_spec.tedinfo->
586:                                  te_ptext[4];
587:
588:    if (flag=='a')
589:    {
590:      /* Adressen für Datei aktivieren */
591:      array=inact;
592:      begin=inactive.begin;
593:      buffer=inactive.buffer;
594:    }
595:    else
596:    {
597:      /* Adressen für Datei deaktivieren */
598:      array=act;
599:      begin=active.begin;
600:      buffer=active.buffer;
601:    }
602:
603:    /* Index des selektierten Eintrags ermitteln */
604:    for (i=0; i<VISIBLE; i++)
605:      if (boot[array[i]].ob_state & SELECTED)
606:        break;
607:    index=begin+i;
608:
609:    /* Pfad je nach Programmtyp erstellen */
610:    path1[0]=boot_device();
611:    strcpy(&path1[1],((active.type==ACC) ? ":\\" :
612:                      ":\\AUTO\\"));
613:    strcat(path1,buffer[index]);
614:
615:    for (i=(WORD)strlen(path1)-1; i>=0; i—)
616:      if (path1[i]!=' ')
617:      {
618:        path1[++i]=EOS;
619:        break;
620:      }
621:
622:    strcat(path1,((active.type==ACC) ?
623:                  ".ACC" : ".PRG"));
624:    strcpy(path2,path1);
625:    path1[strlen(path1)-1]='X';
626:
627:    /* Datei umnennen */
628:    if (flag=='a')
629:    {
630:      if (Frename(0,path1,path2)>=0)
631:        copy(&active,&inactive,index);
632:    }
633:    else
634:    {
635:      if (Frename(0,path2,path1)>=0)
636:        copy(&inactive,&active,index);
637:    }
638:  }
639:
640:
641:  /*************************************************/
642:  /* Umbenannte Datei aus der einen Liste ent-   */
643:  /* fernen und in die andere Liste eintragen    */
644:  /* Übergabeparameter: Zeiger auf Ziel- und     */
645:  /*                    Quell-Deskriptor,        */
646:  /*                    Index des zu entfernen-  */
647:  /*                    den Eintrags             */
648:  /* Rückgabe: keine                             */
649:  /*************************************************/
650:
651:  VOID copy(BD *dest, BD *source, WORD index)
652:  {
653:    WORD i;
654:
655:    /* Eintrag in Ziel-Liste eintragen */
656:    if (dest->num<dest->max_num)
657:      strcpy(dest->buffer[dest->num++],
658:             source->buffer[index]);
659:    else
660:      warning();
661:
662:    /* Eintrag aus der Quell-Liste löschen */
663:    source->num—;
664:    for (i=index; i<source->num; i++)
665:      strcpy(source->buffer[i],
666:             source->buffer[i+1]);
667:    source->begin=dest->begin=0;
668:  }
669:
670:
671:  /*************************************************/
672:  /* Warnung für Listen-Überlauf ausgeben        */
673:  /* Übergabeparameter: keine                    */
674:  /* Rückgabe: keine                             */
675:  /*************************************************/
676:
677:  VOID warning(VOID)
678:  {
                                                    →
```

```
679:   WORD x,y,w,h;
680:
681:   /* Namens-Liste voll */
682:   wind_center(error,&x,&y,&w,&h);
683:   objc_draw(error,ROOT,MAX_DEPTH,x-3,y-3,w+6,h+
       6);
684:   form_do(error,0);
685:   error[MIST].ob_state &= ~SELECTED;
686:   objc_draw(boot,ROOT,MAX_DEPTH,boot[ROOT].ob_x,
687:             boot[ROOT].ob_y,boot[ROOT].ob_width,
688:             boot[ROOT].ob_height);
689: }
690:
691:
692: /*************************************************/
693: /* Boot-Laufwerk ermitteln                       */
694: /* Übergabeparameter: keine                      */
695: /* Rückgabe: Boot-Laufwerk                       */
696: /*************************************************/
697:
698: char boot_device(VOID)
699: {
700:   LONG ssp;
701:   char boot;
702:
703:   ssp=Super((VOID *)0L);
704:   boot=*(BYTE *)_bootdev+'A';
705:   Super((VOID *)ssp);
706:   return(boot);
707: }
708:
709:
710: /*************************************************/
711: /* Im Fenster nach oben scrollen                 */
712: /* Übergabeparameter: Fensterkennung indirekt    */
713: /*                    über scroll_param          */
714: /* Rückgabe: keine                               */
715: /*************************************************/
716:
717: VOID scroll_up(VOID)
718: {
719:   BD *bd=((scroll_param==ACTIVE) ?
720:           &active : &inactive);
721:
722:   if (bd->begin>0)
723:   {
724:     bd->begin--;
725:     into_resource(bd,TRUE);
726:   }
727: }
728:
729:
730: /*************************************************/
731: /* Im Fenster nach unten scrollen                */
732: /* Übergabeparameter: Fensterkennung indirekt    */
733: /*                    über scroll_param          */
734: /* Rückgabe: keine                               */
735: /*************************************************/
736:
737: VOID scroll_down(VOID)
738: {
739:   BD *bd=((scroll_param==ACTIVE) ?
740:           &active : &inactive);
741:
742:   if (bd->begin+VISIBLE<bd->num)
743:   {
744:     bd->begin++;
745:     into_resource(bd,TRUE);
746:   }
747: }
748:
749:
750: /*************************************************/
751: /* Namensliste in die Dialogbox eintragen        */
752: /* Übergabeparameter: Zeiger auf Buffer-         */
753: /*                    Deskriptor, Zeichen-Flag   */
754: /* Rückgabe: keine                               */
755: /*************************************************/
756:
757: VOID into_resource(BD *bd, WORD draw)
758: {
759:   WORD i;
760:   WORD *array=((bd==&active) ? act :inact);
761:
762:   for (i=0; i<VISIBLE; i++)
763:     if (bd->begin+i<bd->num)
764:     {
765:       /* Eintrag vorhanden */
766:       boot[array[i]].ob_spec.tedinfo->te_ptext=
767:         bd->buffer[bd->begin+i];
768:       boot[array[i]].ob_flags=SELECTABLE |
769:         RBUTTON | TOUCHEXIT;
770:       if (bd->begin+i==bd->selected)
771:         boot[array[i]].ob_state |= SELECTED;
772:       else
773:         boot[array[i]].ob_state &= ~SELECTED;
774:     }
775:     else
776:     {
777:       /* kein Eintrag mehr */
778:       boot[array[i]].ob_spec.tedinfo->te_ptext=
779:         underlined;
780:       boot[array[i]].ob_flags=TOUCHEXIT;
781:       boot[array[i]].ob_state &= ~SELECTED;
782:     }
783:
784:   /* Liste neu ausgeben */
785:   if (draw)
786:     redraw_object(boot,(array==act) ?
787:                   ACTWIND : INACWIND);
788: }
789:
790:
791: /*************************************************/
792: /* Vertikalen Slider positionieren               */
793: /* Übergabeparameter: Zeiger auf Dialogbox,      */
794: /*                    Index Slider-Hintergrund,*/
795: /*                    Index Slider, Zeiger auf */
796: /*                    Slider-Position, Buffer- */
797: /*                    Deskriptor                */
798: /* Rückgabe: keine                               */
799: /*************************************************/
800:
801: VOID pos_vslider(OBJECT *tree, WORD parent,
802:                  WORD slider, WORD *slider_pos,
803:                  BD *bd)
804: {
805:   WORD my,y;
806:   WORD dummy;
807:   WORD max,temp;
808:
809:   /* Koordinaten einlesen */
810:   graf_mkstate(&dummy,&my,&dummy,&dummy);
811:   objc_offset(tree,slider,&dummy,&y);
812:
813:   /* je nach Mausposition entweder nach oben
814:      oder unten scrollen */
815:   if (my<y)
816:     *slider_pos=((max=bd->num-VISIBLE)<=
817:                 (temp=*slider_pos+VISIBLE)) ?
818:                 max : temp;
819:   else
820:     *slider_pos=((max=*slider_pos-VISIBLE)<0) ?
821:                 0 : max;
822:
823:   /* Slider neu positionieren */
824:   max=((max=bd->num-VISIBLE)<0) ? 0 : max;
825:   (*params->pos_vslider)(tree,parent,slider,
826:                          *slider_pos,0,max,NIL);
827:
828:   /* Namensliste neu ausgeben */
829:   bd->begin=((max=bd->num-VISIBLE-*slider_pos)<0)
830:             ? 0 : max;
831:   into_resource(bd,TRUE);
832:   redraw_object(tree,parent);
833: }
834:
835:
836: /*************************************************/
837: /* Vertikalen Slider verschieben                 */
838: /* Übergabeparameter: Zeiger auf Dialogbox,      */
839: /*                    Index Slider-Hintergrund,*/
840: /*                    Index Slider, Zeiger auf */
841: /*                    Slider-Position, Buffer- */
842: /*                    Deskriptor                */
843: /* Rückgabe: keine                               */
844: /*************************************************/
845:
846: VOID move_vslider(OBJECT *tree, WORD parent,
847:                   WORD slider, WORD *slider_pos,
848:                   BD* bd)
849: {
850:   WORD max;
851:
852:   max=((max=bd->num-VISIBLE)<0) ? 0 : max;
853:   (*params->move_vslider)(tree,parent,slider,0,
854:                           max,slider_pos,NIL);
855:
856:   /* je nach Slider-Position die Namensliste
857:      neu ausgeben */
858:   bd->begin=((max=bd->num-VISIBLE-*slider_pos)<0)
859:             ? 0 : max;
860:   into_resource(bd,TRUE);
861: }
862:
863:
864: /*************************************************/
865: /* Buffer-Deskriptoren initialisieren            */
866: /* Übergabeparameter: Zeiger auf beide Buffer    */
867: /* Rückgabe: keine                               */
868: /*************************************************/
869:
870: VOID init_bd(char (*act_buff)[NAME_LENGTH],
871:              char (*inact_buff)[NAME_LENGTH])
872: {
873:   active.buffer=act_buff;
874:   active.max_num=MAX_ACTIVE;
875:   active.begin=0;
876:   active.type=ACC;
877:   active.selected=-1;
878:
879:   inactive.buffer=inact_buff;
880:   inactive.max_num=MAX_INACTIVE;
881:   inactive.begin=0;
882:   inactive.type=ACC;
883:   inactive.selected=-1;
884: }
885:
886:
887: /*************************************************/
888: /* Slider-Position und -GröPe initialisieren     */
889: /* Übergabeparameter: Zeiger auf die beiden      */
890: /*                    Sliderpositionen,          */
891: /*                    Zeichenflag                */
892: /* Rückgabe: keine                               */
893: /*************************************************/
894:
895: VOID init_slider(WORD *slider1_pos,
896:                  WORD *slider2_pos, WORD draw)
897: {
898:   WORD max;
899:
900:   /* Slider-GröPe einstellen */
901:   (*params->size_slider)(boot,PARENT1,SLIDER1,
902:                          active.num,VISIBLE,
903:                          VERTICAL,MIN_SIZE);
```

→

```
904:     (*params->size_slider)(boot,PARENT2,SLIDER2,
905:                            inactive.num,VISIBLE,
906:                            VERTICAL,MIN_SIZE);
907:
908:     /* Slider-Position einstellen */
909:     *slider1_pos=((max=active.num-VISIBLE)<0) ?
910:                 0 : max;
911:     (*params->pos_vslider)(boot,PARENT1,SLIDER1,
912:                            *slider1_pos,0,max,NIL);
913:     *slider2_pos=((max=inactive.num-VISIBLE)<0) ?
914:                 0 : max;
915:     (*params->pos_vslider)(boot,PARENT2,SLIDER2,
916:                            *slider2_pos,0,max,NIL);
917:
918:     /* Slider neu zeichnen */
919:     if (draw)
920:     {
921:       redraw_object(boot,PARENT1);
922:       redraw_object(boot,PARENT2);
923:     }
924: }
925:
926:
927: /*************************************************/
928: /* Dialogbox im Fenster zentrieren               */
929: /* Übergabeparameter: Zeiger auf Dialogbox,      */
930: /*                    Koordinaten                */
931: /* Rückgabe: indirekt über Koordinaten           */
932: /*************************************************/
933:
934: VOID wind_center(OBJECT *tree,WORD *x,WORD *y,
935:                               WORD *w,WORD *h)
936: {
937:   tree[ROOT].ob_x=boot[ROOT].ob_x+
938:                 (boot[ROOT].ob_width-
939:                 tree[ROOT].ob_width)/2;
940:   tree[ROOT].ob_y=boot[ROOT].ob_y+
941:                 (boot[ROOT].ob_height-
942:                 tree[ROOT].ob_height)/2;
943:
944:   *x=tree[ROOT].ob_x;
945:   *y=tree[ROOT].ob_y;
946:   *w=tree[ROOT].ob_width;
947:   *h=tree[ROOT].ob_height;
948: }
949:
950:
951: /*************************************************/
952: /* Pulldown-Menü generieren, darstellen und      */
953: /* auswerten.                                    */
954: /* Übergabeparameter: Zeiger auf Slider-         */
955: /*                    Positionen                 */
956: /* Rückgabe: keine                               */
957: /*************************************************/
958:
959: VOID pulldown(WORD *slider1_pos,
960:               WORD *slider2_pos)
961: {
962:   WORD index,checked;
963:   GRECT button_xywh,window_xywh;
964:   char *pull_adr[2];
965:   char pull_buff[2][15];
966:
967:   /* Texte des Menüs in Buffer eintragen */
968:   strcpy(pull_buff[0],"  ");
969:   strcat(pull_buff[0],acc);
970:   strcat(pull_buff[0],"  ");
971:
972:   strcpy(pull_buff[1],"  ");
973:   strcat(pull_buff[1],prg);
974:   strcat(pull_buff[1],"  ");
975:
976:   /* Index des abgehakten Eintrags */
977:   index=(!strcmp(boot[TYP].ob_spec.free_string,
978:         acc) ? 0 : 1);
979:
980:   /* absolute Button-Koordinaten berechnen */
981:   objc_offset(boot,TYP,&button_xywh.g_x,
982:               &button_xywh.g_y);
983:   button_xywh.g_w=boot[TYP].ob_width;
984:   button_xywh.g_h=boot[TYP].ob_height;
985:
986:   /* absolute Koordinaten der Dialogbox
987:      ermitteln */
988:   objc_offset(boot,ROOT,&window_xywh.g_x,
989:     &window_xywh.g_y);
990:   window_xywh.g_w=boot[ROOT].ob_width;
991:   window_xywh.g_h=boot[ROOT].ob_height;
992:
993:   /* Adressen der einzelnen Einträge in das
994:      Übergabe-Array eintragen */
995:   pull_adr[0]=pull_buff[0];
996:   pull_adr[1]=pull_buff[1];
997:
998:   /* Pull-Down-Menü zeichnen lassen und Index des
999:      angeklickten Eintrags zurückliefern */
1000:   checked=(*(params->do_pulldown))
1001:     (pull_adr,2,index,IBM,
1002:     &button_xywh,&window_xywh);
1003:
1004:   /* wenn Eintrag angeklickt wurde... */
1005:   if (checked>=0)
1006:   {
1007:     /* ...dann neuer Eintrag im Button */
1008:     boot[TYP].ob_spec.free_string=
1009:                 ((checked==0) ? acc : prg);
1010:     boot[TYP].ob_state &= ~SELECTED;
1011:     redraw_object(boot,TYP);
1012:
1013:     if (checked!=index)
1014:     {
1015:       /* je nach selektiertem Eintrag neue
1016:          Konfiguration einlesen */
1017:       switch (checked)
1018:       {
1019:         case 0:
1020:           strncpy(activate,"ACC",3);
1021:           strncpy(deactivate,"ACC",3);
1022:           active.type=inactive.type=ACC;
1023:           read_config("*.ACC",&active);
1024:           read_config("*.ACX",&inactive);
1025:           break;
1026:
1027:         case 1:
1028:           strncpy(activate,"PRG",3);
1029:           strncpy(deactivate,"PRG",3);
1030:           active.type=inactive.type=PRG;
1031:           read_config("*.PRG",&active);
1032:           read_config("*.PRX",&inactive);
1033:           break;
1034:       }
1035:
1036:       /* neue Konfiguration anzeigen */
1037:       boot[ACTION].ob_flags=NONE;
1038:       boot[ACTION].ob_spec.tedinfo->te_ptext=
1039:                                       empty;
1040:       redraw_object(boot,ACTION);
1041:
1042:       active.begin=inactive.begin=0;
1043:       active.selected=inactive.selected=-1;
1044:
1045:       into_resource(&active,TRUE);
1046:       into_resource(&inactive,TRUE);
1047:
1048:       init_slider(slider1_pos,slider2_pos,TRUE);
1049:     }
1050:   }
1051: }
```

Listing 5: BOOT.C

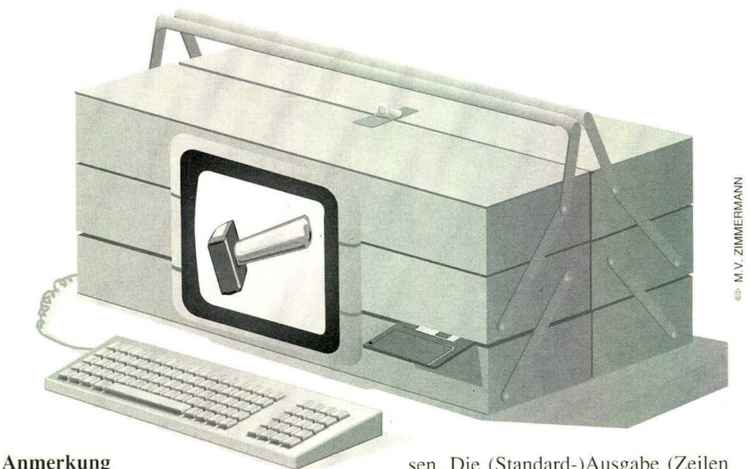
ROCKUS







Programmer's Toolbox - Dateien



Teil 10: Die Kommandos CAT, MORE und GREP

In der heutigen Folge der Programmer's Toolbox befassen wir uns mit den drei folgenden Kommandos zur Bearbeitung von Textdateien:

CAT	- Verschmelzen und Anzeigen von Dateien
MORE	- Anzeigen von Textdateien
GREP	- Durchsuchen von Textdateien nach Textmustern

Wenden wir uns gleich deren Beschreibung bzw. Programmierung zu.

Das Kommando CAT

Name
CAT - Verschmelzen und Anzeigen von Dateien

Anwendung
CAT [-BNS] [*Dateiname...*]

Beschreibung
CAT liest die angegebenen *Dateinamen* und gibt sie auf dem Standardausgabekanal aus. Wird kein *Dateiname* angegeben, gibt CAT den Standardeingabekanal aus. Mit Hilfe von Umlenkungen ist es möglich, Dateien zu verschmelzen.

Beispiel
Das Verschmelzen der Dateien TEST1.-TXT und TEST2.TXT zu TESTERG.TXT:

```
CAT TEST1.TXT TEST2.TXT >
                    TESTERG.TXT
```

Optionen
- **-B** Numerierung der Zeilen. Leerzeilen werden dabei nicht berücksichtigt.
- **-N** Numerierung der Zeilen
- **-S** Mehrere aufeinanderfolgende Leerzeilen werden durch eine einzige Leerzeile ersetzt.

Anmerkung
„-B“ und „-N“ schließen sich wechselseitig aus.

Bei der Anwendung von CAT ist zu beachten, daß die maximale Zeilenlänge den Wert 255 nicht überschreiten darf. Sollen Dateien mit längeren Zeilen bearbeitet werden, ist es erforderlich, die Konstante MAXSTRLEN innerhalb der Implementierung von CAT entsprechend zu erhöhen.

Programmierung
Das Kommando CAT finden Sie in Listing 2.4 programmiert. Es knüpft in gewissem Sinne an die Kommandos des vorherigen Blocks an: Auch CAT benutzt den Argumentexpansionsmechanismus des Moduls EXPAND. Entsprechend besitzt es ebenfalls eine rekursive Funktionsabfolge zum Durchlaufen der von EXPAND vordefinierten Datentypen (Funktionen *cat_flist*, *cat_dir*, *cat_dlist*, Zeilen 86-127). Hier zeigt sich nichts Neues. Lediglich die zusätzlichen Parameter innerhalb der rekursiven Funktionskaskade entsprechen den von CAT benötigten Größen. Innerhalb von *cat_flist* wird der Standardeingabekanal mit dem gewünschten Dateinamen wiedereröffnet. Es folgt ein Aufruf von *acat* (Zeilen 46-84). *acat* kopiert die Standardeingabe auf die Standardausgabe, unter Berücksichtigung der Optionen *num* (Numerierung aller nichtleeren Zeilen), *numall* (Numerierung aller Zeilen) und *subst* (aufeinanderfolgende Leerzeilen werden als eine einzige Leerzeile ausgegeben). Die Ausgabe findet dabei innerhalb einer *while*-Schleife statt (Zeilen 57-83). Die Standardeingabe wird in den Zeilen 55-56 und 81-82 gelesen. Die (Standard-)Ausgabe (Zeilen 58-80) erfolgt unter Berücksichtigung der Parameter von *acat*. Das äußere *if* (Zeile 58) stellt dabei sicher, daß unerwünschte Leerzeilen nicht ausgegeben werden - die inneren Anweisungen werden dann gar nicht erreicht. Im weiteren Ablauf sind Numerierungswünsche zu berücksichtigen. Wenn *num* gewünscht wird, erfolgt die numerierte Ausgabe (Zeilen 60-69). Wird numall verlangt, geschieht die numerierte Ausgabe unter Ignorierung der Leerzeilen (Zeilen 71-75), ansonsten erhält man eine unnumerierte Ausgabe (Zeilen 76-79).

Das Kommando MORE

Name
MORE - Anzeigen von Textdateien

Anwendung
MORE [-FS] [*-Zeilen*] [*+Zeilennummer*] [+/*Muster*] *Dateiname...*

Beschreibung
MORE gibt den Inhalt einer Textdatei seitenweise auf dem Standardausgabekanal aus. Nach der Ausgabe einer Seite wird abgestoppt und auf die Eingabe eines Befehls von der Konsole gewartet. Folgende Befehle sind möglich:

SPACE	Es wird um eine Seite vorwärtsgeblättert.
RETURN	Es wird um eine Zeile vorwärtsgeblättert.
q	Das Kommando MORE wird verlassen.

Werden mehrere Textdateien in der Kommandozeile spezifiziert, wird den Dateien ein Kopf vorangestellt, der den jeweiligen Dateinamen enthält.

Wenn Standardein- oder -ausgabe nicht auf der Konsole erfolgen, geschieht die Textausgabe ohne Unterbrechungen und entspricht damit im wesentlichen dem parameterlosen CAT. Der einzige Unterschied zwischen CAT und MORE besteht in diesem Fall darin, daß MORE beim Auftreten von mehreren Dateien allen Dateien jeweils ihren Dateinamen voranstellt.

Optionen

-F Überlange Zeilen werden abgeschnitten und standardmäßsig zerteilt.

-S Mehrere aufeinanderfolgende Leerzeilen werden durch eine einzige Leerzeile ersetzt.

-Zeilen
Normalerweise werden jeweils 22 Zeilen von MORE auf einer Seite ausgegeben. Mit der Option „*-Zeilen*" wird der Wert auf Zeilen Zeilen.

+Zeilennummer
Starte die Ausgabe erst ab *Zeilennummer*.

+/Muster
Starte die Ausgabe erst beim ersten Auftreten von *Muster*. *Muster* darf das Zeichen '?' zur Markierung eines unbekannten Buchstabens und das Zeichen '*' zur Markierung eines unbekannten Teil-Strings enthalten.

Anmerkung

Die Optionen „*+Zeilennummer*" und „*+/Muster*" setzen sich wechselseitig zurück.

Für die maximale Zeilenlänge gilt das bei CAT Gesagte, d.h. die Zeilen dürfen nicht länger sein als 255 Zeichen. Ansonsten ist MORE.C unter Änderung von MAXSTRLEN zu recompilieren.

Programmierung

Das Kommando MORE wurde in Listing 2.5 programmiert. MORE besitzt recht üppige Möglichkeiten zur Argumentübergabe. Aus diesem Grund soll zunächst die Interpretation der Argumente näher betrachtet werden (Funktion *more*, Zeilen 202-279). Interessant ist dabei die Optionsanalyse (Zeilen 221-256), da hier mehr als nur das einfache Abprüfen von gesetzten bzw. nicht gesetzten Optionen geschieht. MORE erlaubt nämlich zusätzlich zur Angabe von „herkömmlichen" Optionen (führendes Minus und nachfolgender Buchstabe) gewissermaßen parametrisierte Optionen. Hierbei handelt es sich um:

-Zeilen	Die Angabe der Seitengröße.
+Zeilennummer	Die Startzeile für die Ausgabe.
+/Muster	Ein Suchmuster für den Start der Ausgabe.

Die Interpretation der herkömmlichen Optionen geschieht dabei in den Zeilen 222-227, die Angabe der Seitengröße in den Zeilen 228-235, die Angabe einer Startzeile in den Zeilen 243-251. Das Muster wird in den Zeilen 238-242 gelesen. Aus den hier angesiedelten Auswertungen resultiert ein umfangreicher Parametersatz für die rekursive Funktionskaskade von MORE (auch MORE benutzt EXPAND). Im einzelnen handelt es sich dabei um folgende Parameter:

fold	Abschneiden überlanger Zeilen
subst	Ersetzen mehrerer Leerzeilen durch eine einzige Leerzeile
lines	Angabe der Seitengröße
linenum	Angabe der Startzeile mit einer Zeilennummer
pldisc	Diskriminator für die Angabe der Startzeile (0: Angabe mit einer Zeilennummer; 1: Angabe mit einem Muster)
pattern	Angabe der Startzeile mit einem Muster

Am Ende der rekursiven Funktionskaskade steht die Funktion *more_flist* (Zeilen 128-152), die genau wie *cat_flist* beim Kommando CAT das Öffnen des Standardeingabekanals übernimmt und dann *amore* aufruft. Neben den obigen Parametern erzeugt *more_flist* dabei noch zusätzlich den Parameter *mfiles* und übergibt ihn zusammen mit dem Dateinamen an *amore*. In *mfiles* ist die Information enthalten, ob MORE mit einer einzigen Datei zur Pfadexpansion aufgerufen wurde oder ob weitere Dateien vorliegen. Innerhalb von *amore* (Zeilen 66-124) entscheidet *mfiles* darüber, ob die Datei mit einem Dateikopf versehen wird oder nicht. Es folgt eine *while*-Schleife (Zeilen 94-123), die starke Ähnlichkeit mit der entsprechenden *while*-Schleife innerhalb der Funktion *acat* (Listing 2.4) besitzt. Auch in MORE wird diese Schleife benutzt, um den Standardeingabekanal bis zum Dateiende zu durchlaufen und zeilenweise einzulesen. Ob die gelesenen Zeilen angezeigt werden oder nicht, entscheidet sich in den Zeilen 95-99 - hier wird auf die Zeilennummer bzw. das Muster geprüft. Die Ausgabe wird erst beim Auffinden einer entsprechenden Zeile aktiviert. Innerhalb der Ausgabe (Zeilen 100-118) sind nun die übrigen Parameter von *amore* zu berücksichtigen. Zunächst ist dies der Parameter *subst*. Entsprechend wird nur dann eine Ausgabe vorgenommen, wenn die Bedingung in den Zeilen 100-101 erfüllt ist. Immer dann, wenn die Ausgabe auf ein Terminal erfolgt und eine entsprechende Zeilenzahl ausgegeben wurde, ist der Benutzer gefragt, und die Ausgabe der Zeilen hält für eine Benutzereingabe an (Zeile 108). Drei Möglichkeiten sind nun gegeben:

1. Eingabe eines Leerzeichens (Zeile 109)
   => Es wird eine neue Seite ausgegeben.
2. Eingabe des Buchstabens 'q'
   (Zeile 111)
   => Das Kommando MORE wird verlassen.
3. Eingabe eines RETURNs (Zeile 113)
   => Es wird eine neue Zeile ausgegeben.

Alle anderen Eingaben werden ignoriert.

Das Kommando GREP

In Folge der modularen Entwicklung von Programmen kommt es häufig vor, daß etwa bei der Änderung von Modulen ein globaler Bezeichner geändert wird oder wegfällt. In der Regel steht man dann vor dem Problem, daß nicht genau bekannt ist, von welchen anderen Modulen dieser Bezeichner importiert worden ist. Entsprechend ist nicht klar, welche Textstellen geändert werden müssen. Zu diesem Zweck werden vom Betriebssystem UNIX die Funktionen der GREP-Familie bereitgestellt. Sie durchsuchen Textdateien nach bestimmten Zeichenketten bzw. Mustern. Mit dem Kommando GREP soll eine ähnliche Funktion auch für den ST programmiert werden.

Name

GREP - Durchsuchen von Textdateien nach Textmustern

Anwendung

GREP [-CHNV] *Muster Dateiname...*

Beschreibung

Die angegebenen Dateien werden nach dem Textmuster *Muster* durchsucht. *Muster* darf dabei das Zeichen '?' zur Kennzeichnung eines unbekannten Buchstabens und das Zeichen '*' zur Kennzeichnung eines unbekannten Teil-Strings enthalten.

Findet GREP Zeilen, auf die das Muster zutrifft, dann wird der Dateiname gefolgt von der Zeile ausgegeben.

Optionen

-C Pro Datei wird die Anzahl der gefundenen Textstellen anstelle der Textstellen ausgegeben.

-H Die Dateinamen werden nicht ausgegeben.

-N Jeder aufgefundenen Textstelle wird ihre Zeilennummer innerhalb der Datei vorangestellt.

-V Invertierung der Suche. Es werden nur die Zeilen ausgegeben, die **nicht** auf das angegebene *Muster* zutreffen.

Beispiel

Durchsuche sämtliche Textdateien (Endung TXT) nach der Zeichenkette „Mit freundlichen Grüßen". Gebe dabei die Zeilen aus, in denen die Zeichenkette gefunden wurde.

GREP -N „Mit freundlichen Grüßen" *.TXT

Durchsuche sämtliche C-Dateien nach der switch-Anweisung und zähle ihre Anwendung.

GREP -C -H switch *.C

Anmerkung

Auch GREP unterliegt den bereits von CAT und MORE bekannten Einschränkungen in der Zeilenlänge (Maximallänge = 255 Zeichen).

Programmierung

Das Kommando GREP wurde innerhalb von Listing 2.6 programmiert. Es ist zwar recht lang, mit den Erfahrungen aus MORE läßt es sich aber leicht erläutern. Innerhalb der Argumentinterpretation (Funktion *grep*, Zeilen 180-246) geschieht nichts Unbekanntes. Es werden die GREP spezifischen Optionen „-C", „-H", „-N" und „-V" erkannt (Zeilen 199-206). Danach wird das Muster für den Zeilensuchvorgang zugewiesen (Zeile 215). Es folgen die bekannte, horizontale Argumentexpansion (Zeilen 216-231) und der Aufruf der rekursiven Funktionskaskade mit Einstieg bei der Funktion *grep_dlist*. Die rekursive Funktionskaskade wird, entsprechend den Optionen, diesmal mit folgenden zusätzlichen Parametern versehen:

pattern	das Suchmuster
count	Ausgabe der Anzahl übereinstimmender Textstellen anstelle der Textstellen
hide	Ausgabe ohne Kennung des Dateinamens
num	Ausgabe mit Zeilennummern
invert	Ausgabe aller Textstellen, auf die das Muster nicht zutrifft

Am Ende der rekursiven Funktionsaufrufe steht die Funktion *agrep* (Zeilen 69-110), die die Standardeingabe auf das Muster *pattern* und anhand den angegebenen Optionen untersucht. Im Aufbau ähnelt *agrep* stark der Funktion *amore* des vorangegangenen Abschnitts. Sowohl in *agrep* als auch in *amore* wird die Standardeingabe zunächst zeilenweise durchlaufen und dabei daraufhin untersucht, ob das Muster *pattern* auf die betrachtete Zeile zutrifft (Zeilen 86-88). Ist dies der Fall (Zeilen 89-98), erfolgt eine entsprechende Ausgabe (Zeilen 93-97) bzw. ein Zähler wird hochgezählt (Zeile 91).

Vorausschau

Das war's auch schon wieder für heute. In der nächsten Folge beginnen wir mit der Implementierung des Kommandos SORT. Mit ihm wird das Sortieren von Textdateien auf breiter Basis gelöst. SORT erlaubt unter anderem das Sortieren mit mehreren Sortierfeldern und/oder -relationen.

Dirk Brockhaus

```
 1:  /*
 2:   * Listing 2.4, Datei   : cat.c
 3:   * Programm             : CAT - Verschmelzen und
 4:   *                        Anzeigen von Dateien
 5:   *                        (c) MAXON Computer 1991
 6:   * Modifikationsdatum   : 25-Jan-90
 7:   * Abhängigkeiten       : stdio.h, string.h,
                               local.h,
 8:   *                        atom.h, atom2.h,
                               expand.h
 9:   */
10:
11:  #include <stdio.h>
12:  #include <string.h>
13:  #include "local.h"
14:  #include "atom.h"
15:  #include "atom2.h"
16:  #include "expand.h"
17:
18:  #define MAXSTRLEN   256
19:  #define MAXPATHLEN 100
20:
21:  /*
22:   * Funktionen     : acat, cat_flist, cat_dir,
                        cat_dlist
23:   *
24:   * Parameter      : acat(num, numall, subst);
25:   *                  cat_flist(path, flist, num,
26:   *                        numall,subst);
27:   *                  cat_dir(dir, num, numall,
                               subst);
28:   *                  cat_dlist(dlist, num, numall,
                                 subst);
29:   *                  char             *path;
30:   *                  FILE_DESC_LIST flist;
31:   *                  BOOLEAN          num,
32:   *                                   numall,
33:   *                                   subst;
34:   *                  DIR_DESC_ITEM    dir;
35:   *                  DIR_DESC_LIST    dlist;
36:   *
37:   * Aufgabe        :
38:   *
39:   * Durchlauf der mit den Funktionen
       <built_dlist> und
40:   * <expand_dlist> erzeugten Datenstruktur. Die so
41:   * durchlaufenen Dateien werden gemäß den drei
       Optionen
42:   * <num>, <numall> und <subst> auf die
       Standardausgabe
43:   * geschrieben. Dabei wird die Funktion <acat>
       benutzt,
44:   * um die Ausgabe vorzunehmen.
45:   */
46:
47:  void acat(num, numall, subst)
48:  BOOLEAN num,
49:          numall,
50:          subst;
51:  {   long     lnum = 1L;
52:      char     lbuffer[MAXSTRLEN],
53:               *ptr;
54:      BOOLEAN previous = FALSE;
55:
56:      strcpy(lbuffer, "");
57:      ptr = gets(lbuffer);
58:      while(!(strcmp(lbuffer, "") == 0 && ptr ==
             NULL)) {
59:          if (!(subst && previous &&
                 onlyws(lbuffer))) {
60:              if (num) {
61:                  if (onlyws(lbuffer)) {
62:                      printf("      %s\n",
                             lbuffer);
63:                      previous = TRUE;
64:                  }
65:                  else {
66:                      printf("%4ld  %s\n", lnum,
                                 lbuffer);
67:
68:                      previous = FALSE;
69:                      lnum++;
70:                  }
71:              }
72:              else if (numall) {
73:                  printf("%4ld  %s\n", lnum,
                             lbuffer);
74:                  lnum++;
75:                  previous = subst &&
                                onlyws(lbuffer);
76:              }
77:              else {
78:                  printf("%s\n", lbuffer);     →
```

```
 79:                previous = subst &&
                               onlyws(lbuffer);
 80:            }
 81:        }
 82:        strcpy(lbuffer, "");
 83:        ptr = gets(lbuffer);
 84:    }
 85: }
 86:
 87: void cat_flist(path, list, num, numall, subst)
 88: char            *path;
 89: FILE_DESC_LIST list;
 90: BOOLEAN         num,
 91:                 numall,
 92:                 subst;
 93: {   FILE_DESC_LIST work;
 94:     char            filepath[MAXPATHLEN];
 95:
 96:     while (list != NULL) {
 97:         if (!(list->fdesc.fattr & 0x10)) {
 98:             sprintf(filepath, "%s\\%s", path,
 99:                     list->fdesc.fname);
100:             freopen(filepath, "r", stdin);
101:             acat(num, numall, subst);
102:         }
103:         list = list->next;
104:     }
105: }
106:
107: void cat_dir(dir, num, numall, subst)
108: DIR_DESC_ITEM dir;
109: BOOLEAN         num,
110:                 numall,
111:                 subst;
112: {   if (strlen(dir.dirpath) == 3 &&
113:         dir.dirpath[2] == '\\')
114:         dir.dirpath[2] = 0;
115:     cat_flist(dir.dirpath, dir.flist, num,
116:               numall,subst);
117: }
118:
119: void cat_dlist(dlist, num, numall, subst)
120: DIR_DESC_LIST dlist;
121: BOOLEAN         num,
122:                 numall,
123:                 subst;
124: {   if (dlist != NULL) {
125:         cat_dir(*dlist, num, numall, subst);
126:         cat_dlist(dlist->next, num, numall,
                       subst);
127:     }
128: }
129:
130: /*
131:  * Funktion      : cat
132:  *
133:  * Parameter     : cat(argc, argv);
134:  *                 short argc;
135:  *                 char  *argv[];
136:  *
137:  * Aufgabe       :
138:  *
139:  * Interpretation der durch <argc> und <argv>
140:  * spezifizierten Parameterliste gemäß den Fest-
141:  * legungen des Kommandos CAT.
142:  */
143:
144: void cat(argc, argv)
145: short argc;
146: char  *argv[];
147: {   DIR_DESC_LIST dlist;
148:     char            path[MAXPATHLEN];
149:     BOOLEAN         option   = TRUE,
150:                     argument = FALSE,
151:                     start    = TRUE,
152:                     num      = FALSE,
153:                     numall   = FALSE,
154:                     subst    = FALSE;
155:     short           i;
156:
157:     for (i = 1; i < argc; i++) {
158:         convupper(argv[i]);
159:         if (option && argv[i][0] == '-') {
160:             if (strcmp(argv[i], "-B") == 0) {
161:                 num = TRUE;
162:                 numall = FALSE;
163:             }
164:             else if (strcmp(argv[i], "-N") == 0)
                 {
165:                 numall = TRUE;
166:                 num = FALSE;
167:             }
168:             else if (strcmp(argv[i], "-S") == 0)
169:                 subst = TRUE;
170:             else
171:                 fprintf(stderr,
172:                    "Unknown option %s\n",
                        argv[i]);
173:         }
174:         else {
175:             option = FALSE;
176:             argument = TRUE;
177:             if (start) {
178:                 dlist = built_dlist(argv[i],
179:                         FALSE,TRUE, FALSE, 0);
180:                 start = dlist->flist ==
181:                         (FILE_DESC_LIST)ERROR;
182:             }
183:             else
184:                 dlist = expand_dlist(dlist,
                            argv[i],
185:                         FALSE, TRUE, FALSE, 0);
186:         }
187:     }
188:     if (!argument)
189:         acat(num, numall, subst);
190:     else {
191:         if (!start)
192:             cat_dlist(dlist, num, numall, subst);
193:         drop_dlist(dlist);
194:     }
195: }
196:
197: void main(argc, argv)
198: short argc;
199: char  *argv[];
200: {   cat(argc, argv);
201:     exit(0);
202: }
```

```
  1: /*
  2:  * Listing 2.6, Datei    : grep.c
  3:  * Programm              : GREP - Durchsuchen von
                               Dateien
  4:  *                         nach Textmustern
  5:  *                         (c) MAXON Computer 1991
  6:  * Modifikationsdatum    : 30-Jan-90
  7:  * Abhängigkeiten        : stdio.h, string.h,
                               local.h,
  8:  *                         atom.h, atom2.h,
                               expand.h
  9:  */
 10:
 11: #include <stdio.h>
 12: #include <string.h>
 13: #include "local.h"
 14: #include "atom.h"
 15: #include "atom2.h"
 16: #include "expand.h"
 17:
 18: #define MAXSTRLEN  256
 19: #define MAXPATHLEN 100
 20:
 21: /*
 22:  * Funktionen    : agrep, grep_flist, grep_dir,
 23:  *                 grep_dlist
 24:  *
 25:  * Parameter     : agrep(pattern, count, hide,
 26:  *                       num,invert);
 27:  *                 grep_flist(path, flist, num,
 28:  *                            numall,subst);
 29:  *                 grep_dir(dir, pattern, count,
 30:  *                          hide, num, invert);
 31:  *                 grep_dlist(dlist, pattern,
                                  count,
 32:  *                            hide, num, invert);
 33:  *                 char            *path,
 34:  *                                 *pattern;
 35:  *                 FILE_DESC_LIST flist;
 36:  *                 BOOLEAN         count,
 37:  *                                 hide,
```

→

```
 38:  *                                 num,
 39:  *                                 invert;
 40:  *                 DIR_DESC_ITEM   dir;
 41:  *                 DIR_DESC_LIST   dlist;
 42:  *
 43:  * Aufgabe      :
 44:  *
 45:  * Durchlauf der mit den Funktionen
      <built_dlist> und <expand_dlist> erzeugten
 46:  * Datenstruktur. Die durchlaufenen Dateien
 47:  * werden nach Zeilen mit dem Muster <pattern>
 48:  * untersucht. Werden Zeilen gefunden, auf die
 49:  * das Muster <pattern> zutrifft, dann wird eine
 50:  * entsprechende Ausgabe auf der Standardausgabe
 51:  * erzeugt. Im Normalfall beinhaltet diese
 52:  * Ausgabe den Dateinamen und die Zeile mit der
 53:  * Übereinstimmung. Die übrigen Parameter
      besitzen folgende Bedeutung:
 54:  *
 55:  * <count>:
 56:  * Ausgabe der Anzahl der Übereinstimmungen
 57:  * anstelle der betroffenen Zeilen.
 58:  *
 59:  * <hide>:
 60:  * Ausgabe ohne Nennung des Dateinamens.
 61:  *
 62:  * <num>:
 63:  * Ausgabe mit Zeilennummer.
 64:  *
 65:  * <revert>:
 66:  * Ausgabe aller Textstellen, auf die das Muster
 67:  * <pattern> nicht zutrifft.
 68:  */
 69:
 70: void agrep(filepath, pattern, count, hide,
 71:            num, invert)
 72: char    *filepath,
 73:         *pattern;
 74: BOOLEAN count,
 75:         hide,
 76:         num,
 77:         invert;
 78: {   long    lnum = 1L;
 79:     char    lbuffer[MAXSTRLEN],
 80:             *ptr;
 81:     short   number = 0;
 82:     BOOLEAN matching;
 83:
 84:     strcat(lbuffer, "");
 85:     ptr = gets(lbuffer);
 86:     while(!(strcmp(lbuffer, "") == 0 && ptr ==
          NULL)) {
 87:         matching = patmat(pattern, lbuffer);
 88:         if (invert)
 89:             matching = !matching;
 90:         if (matching) {
 91:             if (count)
 92:                 number++;
 93:             else {
 94:                 if (!hide)
 95:                     printf("%s: ", filepath);
 96:                 if (num)
 97:                     printf("%ld ", lnum);
 98:                 printf("%s\n", lbuffer);
 99:             }
100:
101:         }
102:         strcat(lbuffer, "");
103:         ptr = gets(lbuffer);
104:         lnum++;
105:     }
106:     if (count && number > 0) {
107:         if (!hide)
108:             printf("%s: ", filepath);
109:         printf("%d occurence(s) of %s\n", number,
                pattern);
110:     }
111: }
112:
113: void grep_flist(path, list, pattern, count, hide,
114:                 num, invert)
115: char            *path;
116: FILE_DESC_LIST  list;
117: char            *pattern;
118: BOOLEAN         count,
119:                 hide,
120:                 num,
121:                 invert;
122: {   FILE_DESC_LIST work;
123:     char           filepath[MAXPATHLEN];
124:
125:     while (list != NULL) {
126:         if (!(list->fdesc.fattr & 0x10)) {
127:             sprintf(filepath, "%s\\%s", path,
128:                     list->fdesc.fname);
129:             freopen(filepath, "r", stdin);
130:             agrep(filepath, pattern, count, hide,
131:                   num, invert);
132:         }
133:         list = list->next;
134:     }
135: }
136:
137: void grep_dir(dir, pattern, count, hide, num,
              invert)
138: DIR_DESC_ITEM dir;
139: char          *pattern;
140: BOOLEAN       count,
141:               hide,
142:               num,
143:               invert;
144: {   if (strlen(dir.dirpath) == 3 &&
145:         dir.dirpath[2] == '\\')
146:         dir.dirpath[2] = 0;
147:     grep_flist(dir.dirpath, dir.flist, pattern,
148:                count, hide, num, invert);
149: }
150:
151: void grep_dlist(dlist, pattern, count, hide, num,
152:                 invert)
153: DIR_DESC_LIST dlist;
154: char          *pattern;
155: BOOLEAN       count,
156:               hide,
157:               num,
158:               invert;
159: {   if (dlist != NULL) {
160:         grep_dir(*dlist, pattern, count, hide,
161:                  num, invert);
162:         grep_dlist(dlist->next, pattern, count,
163:                    hide, num, invert);
164:     }
165: }
166:
167: /*
168:  * Funktion     : grep
169:  *
170:  * Parameter    : grep(argc, argv);
171:  *                short argc;
172:  *                char  *argv[];
173:  *
174:  * Aufgabe      :
175:  *
176:  * Interpretation der durch <argc> und <argv>
177:  * spezifizierten Parameterliste gemäP den Fest-
178:  * legungen des Kommandos GREP.
179:  */
180:
181: BOOLEAN grep(argc, argv)
182: short argc;
183: char  *argv[];
184: {   DIR_DESC_LIST dlist;
185:     char          path[MAXPATHLEN],
186:                   pattern[MAXSTRLEN];
187:     BOOLEAN       option   = TRUE,
188:                   argument = FALSE,
189:                   start    = TRUE,
190:                   count    = FALSE,
191:                   hide     = FALSE,
192:                   num      = FALSE,
193:                   invert   = FALSE;
194:     short         i;
195:
196:     i = 1;
197:     while (i < argc) {
198:         if (option && argv[i][0] == '-') {
199:             convupper(argv[i]);
200:             if (strcmp(argv[i], "-C") == 0)
201:                 count = TRUE;
202:             else if (strcmp(argv[i], "-H") == 0)
203:                 hide = TRUE;
204:             else if (strcmp(argv[i], "-N") == 0)
205:                 num = TRUE;
```

→

```
206:              else if (strcmp(argv[i], "-V") == 0)
207:                  invert = TRUE;
208:              else
209:                  fprintf(stderr,
210:                      "Unknown option %s\n",
                           argv[i]);
211:          }
212:          else {
213:              option = FALSE;
214:              argument = TRUE;
215:              if (start) {
216:                  strcpy(pattern, argv[i]);
217:                  i++;
218:                  if (i < argc) {
219:                      dlist = built_dlist(argv[i],
220:                              FALSE, TRUE, FALSE, 0);
221:                      start = dlist->flist ==
222:                              (FILE_DESC_LIST)ERROR;
223:                  }
224:                  else {
225:                      fprintf(stderr,
                                 "SYNOPSIS:\n%s\n",
226:                        "GREP [ -CHNV ] pattern
                            filename...");
227:                      return(FALSE);
228:                  }
229:              }
230:              else
231:                  dlist = expand_dlist(dlist,
232:                          argv[i],FALSE, TRUE,
                              FALSE, 0);
233:          }
234:          i++;
235:      }
236:      if (!start) {
237:          grep_dlist(dlist, pattern, count, hide,
238:                     num, invert);
239:          drop_dlist(dlist);
240:          return(TRUE);
241:      }
242:      else {
243:          fprintf(stderr, "SYNOPSIS:\n%s\n",
244:                  "GREP [ -CHNV ] pattern
                       filename...");
245:          return(FALSE);
246:      }
247:  }
248:
249:  void main(argc, argv)
250:  short argc;
251:  char  *argv[];
252:  {   if (!grep(argc, argv))
253:          exit(1);
254:      exit(0);
255:  }
```

```
 1:  /*
 2:   * Listing 2.5, Datei   : more.c
 3:   * Programm             : MORE - Anzeigen von
 4:   *                        Textdateien
 5:   *                        (c) MAXON Computer 1991
 6:   * Modifikationsdatum   : 29-Jan-90
 7:   * Abhängigkeiten       : stdio.h, string.h,
                               osbind.h,
 8:   *                        local.h, atom.h,
                               atom2.h,
 9:   *                        expand.h
10:   */
11:
12:  #include <stdio.h>
13:  #include <string.h>
14:  #include <osbind.h>
15:  #include "local.h"
16:  #include "atom.h"
17:  #include "atom2.h"
18:  #include "expand.h"
19:
20:  #define MAXSTRLEN  256
21:  #define MAXPATHLEN 100
22:
23:  /*
24:   * Funktionen    : amore, more_flist, more_dir,
                       more_dlist
25:   *
26:   * Parameter     : amore(filepath, mfiles, fold,
                             subst,
27:   *                       lines, linenum, pldisc,
                             pattern);
28:   *                 more_flist(path, list, fold,
                             subst,
29:   *                       lines, linenum, pldisc,
                             pattern);
30:   *                 more_dir(dir, fold, subst,
31:   *                       lines, linenum, pldisc,
                             pattern);
32:   *                 more_dlist(dlist, fold, subst,
33:   *                       lines, linenum, pldisc,
                             pattern);
34:   *                 char            *filepath;
35:   *                 BOOLEAN         mfiles,
36:   *                                 fold,
37:   *                                 subst;
38:   *                 short           lines,
39:   *                                 linenum,
40:   *                                 pldisc;
41:   *                 char            *pattern,
42:   *                                 *path;
43:   *                 FILE_DESC_LIST list;
44:   *                 DIR_DESC_ITEM  dir;
45:   *                 DIR_DESC_LIST  dlist;
46:   *
47:   * Aufgabe       :
48:   *
49:   * Durchlauf der mit den Funktionen
        <built_dlist> und
50:   * <expand_dlist> erzeugten Datenstruktur. Die so
51:   * durchlaufenen Dateien werden gemäP den drei
        Optionen
52:   * <mfiles>, <fold>, <subst> und <lines>
        durchblättert.
53:   * Der Wahrheitswert <mfiles> gibt dabei an, ob
        ein
54:   * Dateikopf mit dem Dateinamen gedruckt werden
        soll.
55:   * <fold> gibt an, daP überlange Zeilen getrennt
        werden.
56:   * Standardmässig werden sie abgeschnitten.
        <subst> gibt
57:   * an, ob mehrere aufeinanderfolgende Leerzeilen
        durch
58:   * eine einzige Leerzeile ersetzt werden und
        <lines> ist
59:   * die jeweilige GröPe der Ausgabeseite (in
        Zeilen).
60:   * <pldisc> ist ein Diskriminator, der darüber
        entscheidet,
61:   * ob der Ausgabebeginn gemäP dem ersten
        Vorkommen des
62:   * Musters <pattern> eingeleitet wird (<pldisc>
        hat den
63:   * Wert 1), oder ob ab der <linenum>-ten Zeile
        mit der
64:   * Ausgabe begonnen wird (<pldisc> hat den Wert
        0.).
65:   */
66:
67:  void amore(filepath, mfiles, fold, subst,
68:             lines, linenum, pldisc, pattern)
69:  char          *filepath;
70:  BOOLEAN       mfiles,
71:                fold,
72:                subst;
73:  short         lines,
74:                linenum,
75:                pldisc;
76:  char          *pattern;
77:  {   long     lnum = 1L;
78:      short    lcount = 0;
79:      char     lbuffer[MAXSTRLEN],
80:               *ptr,
81:               ch;
82:      BOOLEAN previous = FALSE,
83:              output   = FALSE;
84:
85:      if (mfiles) {
86:          printf(
87: "###################################
     ##########\n");
88:          printf("###    %s\n", filepath);
89:          printf(
90: "###################################
```

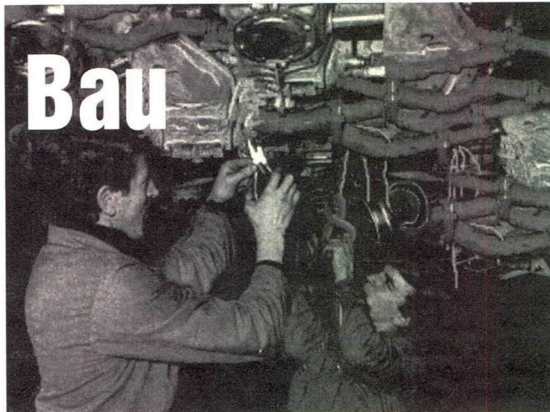
→

```
         ###########\n");
 91:          lcount = 3;
 92:      }
 93:      strcpy(lbuffer, "");
 94:      ptr = gets(lbuffer);
 95:      while(!(strcmp(lbuffer, "") == 0 && ptr ==
             NULL)) {
 96:          if (!output)
 97:              if (pldisc == 0)
 98:                  output = lnum >= linenum;
 99:              else
100:                  output = patmat(pattern,
                                  lbuffer);
101:          if (output &&
102:              !(subst && previous &&
                 onlyws(lbuffer))) {
103:              outline(lbuffer, fold);
104:              previous = onlyws(lbuffer);
105:              lcount++;
106:              if (isatty(fileno(stdout)) &&
107:                  lcount == lines) {
108:                  do {
109:                      ch = Cnecin();
110:                      if (ch == ' ' || ch == 0)
111:                          lcount = 0;
112:                      else if (ch == 'q')
113:                          exit(0);
114:                      else if (ch == 13)
115:                          lcount-;
116:                  }
117:                  while (!(ch == ' ' || ch == 0 ||
118:                           ch == 13));
119:              }
120:          }
121:          strcpy(lbuffer, "");
122:          ptr = gets(lbuffer);
123:          lnum++;
124:      }
125:  }
126:
127:  static BOOLEAN mfiles = FALSE;
128:
129:  void more_flist(path, list, fold, subst,
130:                  lines, linenum, pldisc, pattern)
131:  char            *path;
132:  FILE_DESC_LIST list;
133:  BOOLEAN         fold,
134:                  subst;
135:  short           lines,
136:                  linenum,
137:                  pldisc;
138:  char            *pattern;
139:  {   FILE_DESC_LIST work;
140:      char            filepath[MAXPATHLEN];
141:
142:      while (list != NULL) {
143:          if (!(list->fdesc.fattr & 0x10)) {
144:              sprintf(filepath, "%s\\%s", path,
145:                      list->fdesc.fname);
146:              freopen(filepath, "r", stdin);
147:              mfiles = mfiles || list->next !=
                         NULL;
148:              amore(filepath, mfiles, fold, subst,
149:                    lines, linenum, pldisc,
                      pattern);
150:          }
151:          list = list->next;
152:      }
153:  }
154:
155:  void more_dir(dir, fold, subst,
156:                lines, linenum, pldisc, pattern)
157:  DIR_DESC_ITEM dir;
158:  BOOLEAN       fold,
159:                subst;
160:  short         lines,
161:                linenum,
162:                pldisc;
163:  char          *pattern;
164:  {   if (strlen(dir.dirpath) == 3 &&
165:          dir.dirpath[2] == '\\')
166:          dir.dirpath[2] = 0;
167:      more_flist(dir.dirpath, dir.flist,
168:                 fold, subst, lines, linenum,
169:                 pldisc, pattern);
170:  }
171:
172:  void more_dlist(dlist, fold, subst,
173:                  lines, linenum, pldisc, pattern)
174:  DIR_DESC_LIST dlist;
175:  BOOLEAN         fold,
176:                  subst;
177:  short           lines,
178:                  linenum,
179:                  pldisc;
180:  char            *pattern;
181:  {   if (dlist != NULL) {
182:          more_dir(*dlist, fold, subst, lines,
                       linenum,
183:                   pldisc, pattern);
184:          more_dlist(dlist->next, fold, subst,
185:                     lines, linenum, pldisc, pattern);
186:      }
187:  }
188:
189:  /*
190:   * Funktion      : more
191:   *
192:   * Parameter     : more(argc, argv);
193:   *                 short argc;
194:   *                 char  *argv[];
195:   *
196:   * Aufgabe       :
197:   *
198:   * Interpretation der durch <argc> und <argv>
199:   * spezifizierten Parameterliste gemäP den Fest-
200:   * legungen des Kommandos MORE.
201:   */
202:
203:  void more(argc, argv)
204:  short argc;
205:  char  *argv[];
206:  {   DIR_DESC_LIST dlist;
207:      char          path[MAXPATHLEN];
208:      BOOLEAN       option   = TRUE,
209:                    argument = FALSE,
210:                    start    = TRUE,
211:                    fold     = FALSE,
212:                    subst    = FALSE;
213:      short         lines    = 23,
214:                    linenum  = 1,
215:                    pldisc   = 0;
216:      char          pattern[MAXSTRLEN];
217:      short         i;
218:
219:      strcpy(pattern, "");
220:      for (i = 1; i < argc; i++) {
221:          if (option) {
222:              if (argv[i][0] == '-') {
223:                  if (strcmp(argv[i], "-F") == 0 ||
224:                      strcmp(argv[i], "-f") == 0)
225:                      fold = TRUE;
226:                  else if (strcmp(argv[i], "-S")
                               == 0 ||
227:                           strcmp(argv[i], "-s")
                                  == 0)
228:                      subst = TRUE;
229:                  else {
230:                      lines = -atoi(argv[i]);
231:                      if (lines <= 0) {
232:                          fprintf(stderr,
233:                             "Unknown option %s\n",
                                argv[i]);
234:                          lines = 23;
235:                      }
236:                  }
237:              }
238:              else if (argv[i][0] == '+') {
239:                  if (strlen(argv[i]) >= 2 &&
                          argv[i][1] == '/') {
240:                      argv[i] += 2;
241:                      strcpy(pattern, argv[i]);
242:                      pldisc = 1;
243:                  }
244:                  else {
245:                      linenum = atoi(++argv[i]);
246:                      pldisc = 0;
247:                      if (linenum <= 0) {
248:                          fprintf(stderr,
249:                             "Unknown option %s\n",
                               -argv[i]);
250:                          linenum = 1;
251:                      }
252:                  }
```

→

```
253:                }
254:                else {
255:                    option = FALSE;
256:                    argument = TRUE;
257:                }
258:            }
259:            if (argument) {
260:                if (start) {
261:                    dlist = built_dlist(argv[i],
                                           FALSE,
262:                                       TRUE, FALSE,
                                           0);
263:                    start = dlist->flist ==
264:                        (FILE_DESC_LIST)ERROR;
265:                }
266:                else
267:                    dlist = expand_dlist(dlist,
                                      argv[i],
268:                                  FALSE, TRUE, FALSE,
                                      0);
269:            }
270:        }
271:        if (!argument)
272:            amore("", mfiles, fold, subst,
273:                  lines, linenum, pldisc, pattern);
274:        else {
275:            if (!start)
276:                more_dlist(dlist, fold, subst,
277:                           lines, linenum, pldisc,
                               pattern);
278:            drop_dlist(dlist);
279:        }
280:    }
281:
282:    void main(argc, argv)
283:    short argc;
284:    char  *argv[];
285:    {   more(argc, argv);
286:        exit(0);
287:    }
```

Compiler-Bau



Die zentrale Phase eines Compilers ist dessen semantische Analyse. Sie ermittelt aus dem von Scanner und Parser aufbereiteten Quelltext die Information, die nötig ist, um das Programm korrekt zu übersetzen. Da diese Analyse sehr vielfältig und komplex ist, ist ihr diese Folge komplett gewidmet. Als fließender Übergang zur Codeerzeugung wird am Ende noch gezeigt, wie man mit den Mitteln der semantischen Analyse einen Zwischencode erzeugen kann.

Teil 4

Die semantische Analyse bildet den Abschluß der Analysephase eines Compilers. Sie ermittelt anhand des Strukturbaums einen Teil der Bedeutung des zu übersetzenden Programms. Der Compiler kümmert sich dabei nur um die statische Semantik. Diese kann im Gegensatz zur dynamischen ermittelt werden, ohne daß das Programm ausgeführt wird. Zur statischen Semantik gehören bei den meisten von Compilern übersetzten Programmiersprachen Sachen wie der Bindungsbereich einzelner Deklarationen und die Typen der einzelnen Variablen, Funktionen und Prozeduren. Ein Beispiel für die dynamische Semantik ist etwa die Tatsache, ob ein Programm für eine bestimmte Eingabe terminiert oder in einer Endlosschleife hängen bleibt.

Die semantische Analyse wird meistens durch die Berechnung sogenannter Attribute realisiert, die an die einzelnen Knoten des Strukturbaums gehängt werden. In der Literatur wird dieser Vorgang gerne als Dekorieren des Strukturbaums bezeichnet, was die Assoziation zu einem Weihnachtsbaum recht nahelegt. Welcher Schmuck oder besser, welche Attribute dabei verwendet werden und auf welche Art man sie berechnet, werden wir uns in dieser Folge als erstes ansehen. Danach werden wir eine Art Beschreibungssprache kennenlernen, die es erlaubt, die semantische Analyse eines Compilers recht übersichtlich zu formulieren.

Bei vielen Compilern ist es schwer, eine Grenze zwischen der semantischen Analyse und der ersten Phase der Zwischencode- oder gleich der endgültigen Codeerzeugung zu ziehen. Deshalb werden wir uns gegen Ende dieser Folge kurz ansehen, wie man mit den Mitteln der semantischen Analyse einen Zwischencode erzeugen kann. In der nächsten Folge sehen wir dann, wie man mit diesem Verfahren sofort den Maschinencode erzeugt, ohne den Umweg über den Zwischencode zu nehmen. Zum Abschluß dieser Folge werfen wir wie immer einen Blick auf die Compilercompiler.

Anschlüsse

Nach diesem kurzen Überblick sehen wir uns jetzt die Position an, die die semantische Analyse innerhalb eines kompletten Compilers hat. Gleichzeitig führen wir uns die Schnittstellen zu den anderen Phasen vor Augen. Im folgenden meinen wir sowohl Zwischencode als auch endgültige Codeerzeugung, sobald wir allgemein von Codeerzeugung sprechen.

Die semantische Analyse wird zwischen dem Parser und der Codeerzeugung angesiedelt. Wie wir später noch etwas genauer sehen werden, kann sie dabei sogar mit einer oder beiden angrenzenden Phasen verschmelzen. Wichtig ist dabei, daß diese Verschmelzung nur in der Implementation, nicht aber in dem zu implementierenden Konzept vorgenommen wird. Dies ist sehr wichtig, da die semantische Analyse in der Regel die konzeptionell schwierigste Phase ist. Man kann sogar sagen, daß Scanner und Parser den Quelltext nur für die semantische Analyse aufbereiten und die Codeerzeugung den Code lediglich anhand der Informationen der semantischen Analyse generiert. Die eigentliche Arbeit, nämlich das Umwandeln des aufbereiteten Quelltextes in die Information, die zur Erzeugung des Codes wichtig ist, wird von der semantischen Analyse gemacht.

Die Datenstrukturen, die an den beiden Schnittstellen übergeben werden, sind einmal der vom Parser erzeugte und zum anderen der attributierte Strukturbaum, der an die Codeerzeugung übergeben wird. Beide Bäume können je nach Implementation entweder komplett oder Knoten für Knoten übergeben werden. Eine weitere wichtige Schnittstelle besteht zwischen der semantischen Analyse und der Fehlerbehandlung. Nahezu alle Fehler, bis auf die rein grammatikalischen, werden von der semantischen Analyse erkannt. Das erscheint auch recht logisch, denn die semantische Analyse ermittelt ja einen Teil der Bedeutung des Quelltextes und erkennt Fehler, sobald diese Bedeutung nicht schlüssig ist. Wie schon beim Parser, so ist es auch hier wichtig, nicht nach dem ersten erkannten Fehler abzubrechen, sondern diesen provisorisch zu reparieren und den Übersetzungsvorgang solang wie möglich weiterzuführen. Dies erreicht man in der Regel am besten, indem beim Auftreten

eines Fehlers die zu berechnende Information durch einen speziellen Wert ersetzt wird. Dieser Wert zeigt nachfolgenden Berechnungen an, daß an dieser Stelle ein Fehler aufgetreten ist und der Wert somit ignoriert werden soll, ohne daß ein weiterer Fehler ausgelöst wird. Dieses Verfahren unterdrückt die in der ersten Folge angesprochenen Fehlerlawinen.

Wie schon in den anderen Folgen, wollen wir uns zuerst einmal ansehen, welche Probleme wir dieses Mal zu lösen haben. In Abb. 1 sehen wir den abstrakten Strukturbaum für *x := y + a[i]*. An den Blättern des Baumes hängen die Bezeichnernamen. Natürlich nicht als Zeichenkette, sondern als je ein Knoten, der einen Bezeichner darstellt und einen Verweis in die Bezeichnertabelle enthält. Dieser Verweis entspricht dem Attribut, das das zugehörige Bezeichner-Token besaß. In der Abbildung sind die Bezeichner der Übersicht halber direkt mit Namen eingetragen. Die Symbole ':=', '+' und '[ ]' entsprechen den Strukturbaumknoten für eine Zuweisung, eine Addition und einen Array-Zugriff. Unter dem Strukturbaum ist der Zwischencode zu sehen, der aus dem Baum erzeugt werden soll. In dem Zwischencode wurde der ursprüngliche arithmetische Ausdruck, der aus einem Array-Zugriff, einer Addition und einer Zuweisung besteht, in eben diese drei Bestandteile zerlegt. Außerdem wurde deutlich gemacht, daß es sich um eine Integer-Addition handelt. Diese Zerlegung ist nötig, da es in der Zielsprache (z.B.: 680x0 Assembler) keine Möglichkeit gibt, eine derart komplexe Aktion in einem Befehl auszuführen. Genau genommen muß eine Anweisung, wie der Zugriff auf ein Element eines Arrays, auch noch weiter zerlegt werden. Dies wird aber noch nicht auf der Ebene des Zwischencodes getan, um sich die Möglichkeit einiger Optimierungen offenzuhalten. Wie stark der Zwischencode die Aktionen eines Programms zerlegen soll, hängt etwas von der zu übersetzenden Programmiersprache und sehr stark von der Zielsprache und den Optimierungen ab, die mit Hilfe des Zwischencodes durchgeführt werden sollen.

Auf den ersten Blick könnte man meinen, daß der Strukturbaum ohne weitere Analyse direkt in den Zwischencode umgewandelt werden kann. Für einen Menschen trifft dies auch sicherlich zu, da er die notwendige semantische Analyse mal so nebenbei im Kopf durchführt. Für den Computer und damit einen Compiler ist die Sache nicht so einfach. Der gesamte Ausdruck ist zum Beispiel völliger Unsinn, wenn *a* gar kein Array, sondern eine Integer-Konstante ist, oder wenn *i* eine Fließkommazahl darstellt. Genauso unsinnig wird es, falls *a* ein Array aus Fließkommazahlen ist, aber *y* eine Zeichenkette enthält. In diesem Zusammenhang fällt auch auf, daß der Compiler erst einmal herausfinden muß, ob das '+' zwei Fließkomma- oder zwei ganze Zahlen addieren soll.

Allein für dieses kleine Beispiel muß die semantische Analyse drei verschiedene Aufgaben bewältigen. Erstens muß zu jedem der Bezeichner die zugehörige Deklaration gefunden werden, und dabei muß der Compiler berücksichtigen, daß ein lokal definierter Bezeichner möglicherweise einen globalen Bezeichner gleichen Namens verdeckt. Diese Aufgabe wird als Namensanalyse bezeichnet. Zweitens muß ein Typcheck durchgeführt werden, der sicherstellt, daß die angewendeten Operationen wie Addition und Zugriff auf ein

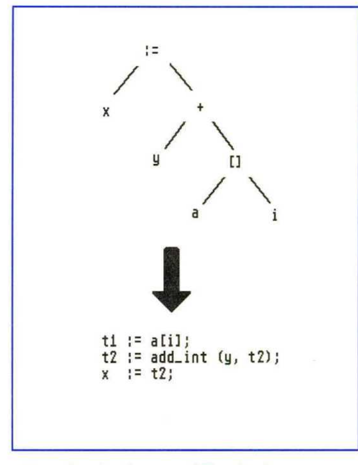


Abb. 1: Strukturbaum und Tupelcode für 'x := y + a[i]'

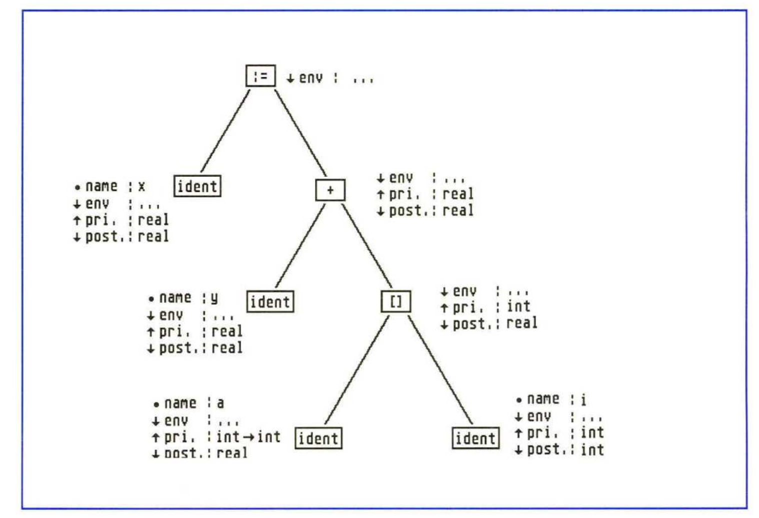


Abb. 2: Attributierter Strukturbaum für 'x := y + a[i]'

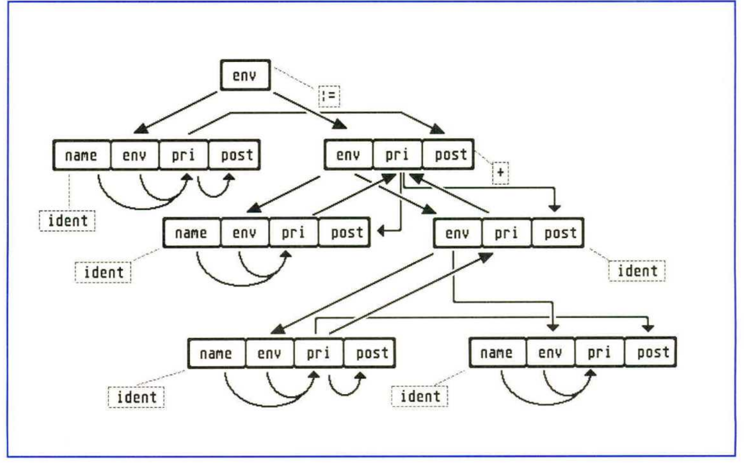


Abb. 3: Abhängigkeiten der Attribute

Array für die benutzten Bezeichner überhaupt zulässig sind. Eng damit verknüpft muß drittens die Überladung von Operatoren aufgelöst werden.

In unserem Beispiel ist genau genommen nicht nur die Addition, sondern auch die Zuweisung überladen. Mit dem gleichen Operator (':=') können sowohl Integer- und Fließkommazahlen als auch ganze Arrays oder Pointer zugewiesen werden. Diese drei Aufgaben müssen neben ein paar anderen in nahezu jedem Compiler durchgeführt werden. Für viele Programmiersprachen ist allerdings noch eine sehr viel weitergehende Analyse notwendig.

Lametta

Bevor wir uns nun genauer damit beschäftigen, wie die semantische Analyse funktioniert, wollen wir erst einmal ihr wesentliches Hilfsmittel kennenlernen. Da es in der semantischen Analyse hauptsächlich darum geht, neue Informationen über die einzelnen Knoten im Strukturbaum zu berechnen und dies nur schrittweise geht, liegt es nahe, diese Information in den einzelnen Knoten des Baumes zu speichern. Das heißt, die Blätter, die Bezeichner repräsentieren, speichern die zu ihnen gehörige Deklaration. Die Operatoren speichern die verarbeiteten Typen, und, falls sie überladen sind, auch noch ihre genaue Funktion. Diese zusätzliche Information wird wie Lametta an die einzelnen Zweige des Strukturbaums gehängt. Allerdings glänzt es in diesem Fall nicht nur, sondern ist für den weiteren Übersetzungsvorgang von entscheidender Bedeutung. Die Werte, die in einen Knoten des Strukturbaums geschrieben werden, heißen dessen Attribute.

Da viele der Attribute voneinander abhängen, muß bei ihrer Berechnung eine bestimmte Reihenfolge eingehalten werden, die stark von der zu analysierenden Programmiersprache abhängt. Die Funktion und das Zusammenspiel der einzelnen Attribute wollen wir uns nun an unserem Beispiel etwas genauer ansehen. In Abb. 2 ist der Strukturbaum für $x := y + a[i]$ noch einmal abgebildet. Diesmal allerdings zusammen mit den nötigen Attributen.

In der Abbildung wurde die semantische Analyse auf vier verschiedene Attribute beschränkt. Das Attribut *env* stellt die aktuelle Umgebung (environment) der Zuweisung dar. Die Umgebung wird im wesentlichen durch den momentanen Inhalt und Zustand der Definitionstabelle bestimmt und enthält Dinge wie die aktuelle Deklaration von Variablen, Prozeduren usw. Das Attribut *name* enthält den Namen eines Bezeichners. Der Name ist allerdings nicht die Zeichenkette, durch die der Bezeichner im Quelltext repräsentiert wurde, sondern die Kennung, die ihm die Bezeichnertabelle verliehen hat. Sie dient dazu, in der aktuellen Umgebung (Definitionstabelle) die Deklaration des Bezeichners zu finden, die bei seinem Auftreten gültig ist. Die beiden Attribute *primode* und *postmode* stehen für den a priori- bzw. a posteriori-Typ des Knotens, an dem sie hängen. In der Abbildung sind sie aus Platzgründen durch *pri* bzw. *post* abgekürzt. Der *primode* gibt den Typ eines Knotens an, der sich aus dem Knoten selber und seinen Nachfolgern ergibt. Der *postmode* gibt hingegen an, welcher Typ von dem Vorgängerknoten erwartet wird. Wozu wir diese beiden Typen brauchen und wie sie berechnet werden, werden wir später noch sehen.

Das Symbol vor jedem Attribut kennzeichnet dessen Ursprung. Der Punkt steht für sogenannte intrinsische Attribute. Diese Attribute sind von Anfang an im Strukturbaum enthalten und stammen vom Scanner. Es sind normalerweise genau die Attribute, die vom Scanner an die Tokens gehängt wurden (siehe auch Folge 2). In unserem Beispiel sind nur die Namen der Bezeichner intrinsisch. Sie wurden vom Scanner ermittelt, als dieser die Bezeichner in die Bezeichnertabelle eingetragen hat, und dann mit dem Token an den Parser weitergegeben, der sie beim Aufbau des Strukturbaums sofort in den Bezeichnerknoten geschrieben hat.

Ein nach unten zeigender Pfeil kennzeichnet Attribute, die im Baum von oben nach unten transportiert und als vererbte Attribute bezeichnet werden. Ein Beispiel ist etwa das Umgebungsattribut *env*, das sukzessive von der Wurzel unseres Beispielausdrucks (':=') bis in alle Blätter (die Bezeichnerknoten) transportiert wird. Das Gegenteil, nämlich die synthetisierten Attribute, sind durch einen nach oben gerichteten Pfeil markiert. Sie werden aus Attributen berechnet, die in demselben Knoten oder dessen Nachfolgern stehen. Im Beispiel ist dies nur das Attribut *postmode*.

Zusammenfassend kann man sagen, daß die intrinsischen Attribute die Information eines Knotens enthalten, die an der entsprechenden Stelle direkt aus dem Quelltext entnommen und vom Scanner an den Parser geliefert wurde. Die vererbten Attribute beinhalten die Information, die der Knoten von seiner Umgebung erhalten hat, also im Quelltext weiter vorne oder hinten steht. Schlußendlich umfassen die synthetisierten Attribute die Information, die in dem Knoten selber oder einem seiner Nachfolger berechnet wurde, also aus der näheren Umgebung stammt.

Diese Klassifizierung erscheint auf den ersten Blick vielleicht etwas willkürlich, ist aber später, beim Bestimmen der Berechnungsreihenfolge für die einzelnen Attribute, sehr nützlich.

Nachdem wir jetzt etwas Struktur in die Information gebracht haben, die von der semantischen Analyse verarbeitet werden soll, sehen wir uns jetzt an, wie man die einzelnen Attribute berechnet und vor allem, wie man diese Berechnungen beschreibt.

Selbst in unserem kleinen Beispiel besitzt jeder Knoten synthetisierte und vererbte Attribute. Außerdem kann ein Attribut nicht berechnet werden, bevor die Attribute bekannt sind, die seine Berechnungsgrundlage bilden. Im Beispiel kann das Umgebungsattribut *env* des Bezeichnerknotens ganz rechts unten nicht berechnet werden, bevor das Umgebungsattribut des Vorgängerknotens '[ ]' nicht bekannt ist. Um den *primode* des Additionsknotens '+' zu berechnen, müssen der *primode* des rechten und linken Nachfolgers bekannt sein. Um etwas Überblick über diese Abhängigkeiten zu gewinnen, sind sie in Abb. 3 grafisch dargestellt.

Ist ein Attribut A von einem zweiten Attribut B abhängig, so ist dies in der Abbildung durch einen Pfeil von B nach A dargestellt. Der Pfeil vom Attribut *env* des mit '[ ]' bezeichneten Knotens zum Attribut *env* des Knotens ganz rechts unten beschreibt also die Abhängigkeit, von der wir gerade gesprochen haben.

Aus der Grafik ist sofort ersichtlich, daß das Attribut *env* gleichmäßig von oben nach unten vererbt wird. Dies ist notwendigerweise der erste Schritt der Attributberechnung, da alle anderen Attribute direkt oder indirekt von *env* abhängen. Die einzige Ausnahme bildet das intrinsische Attribut *name*, das für die Berechnungsreihenfolge uninteressant ist, da es schon von Anfang an im Baum steht. Sobald das Attribut *env* in jedem Bezeichnerknoten bekannt ist, kann daraus das Attribut *primode* von unten nach oben berechnet werden. Zum Schluß kann aus den *primode*-Attributen von oben nach unten das Attribut *postmode* abgeleitet werden.

Für die komplette Berechnung werden somit maximal zwei Baumdurchläufe gebraucht. Im ersten Durchlauf wird zuerst *env* von oben nach unten vererbt und beim Aufsteigen *primode* berechnet. Im zweiten Durchlauf wird dann von oben nach unten laufend *postmode* ermittelt. Außerdem können dabei noch Tests auf die korrekte Verwendung der Typen durchgeführt werden. Zum Beispiel muß *i* vom Indextyp des Arrays *a* sein.

Regeln

Nachdem wir nun etwas Ordnung in die Reihenfolge der Berechnungen gebracht haben, nehmen wir die in jedem Knoten des Baums notwendigen Berechnungen etwas genauer unter die Lupe. Um die Berechnungen übersichtlich zu notieren, werden oft attributierte Grammatiken benutzt. Diese bestehen aus einer Grammatik für den abstrakten Strukturbaum, ähnlich wie wir sie in der letzten Folge kennengelernt haben. Die einzelnen Regeln der Grammatik werden um die Berechnungsregeln erweitert, die zu dem in der Regel beschriebenen Baumknoten gehören.

Listing 1 enthält eine attributierte Grammatik, die auf unser Beispiel paßt. Jeder Grammatikregel, die durch das Schlüsselwort *RULE* gekennzeichnet ist, ist eine Reihe von Attributierungsregeln und Bedingungen zugeordnet, die durch *ATTRIBUTION* bzw. *CONDITION* eingeleitet werden. Die abgebildete Grammatik ist in der Lage, beliebige Zuweisungen zu attributieren, deren rechte Seite aus einer Kombination von Additionen, Array-Zugriffen und Bezeichnern besteht.

Die Grammatikregeln bilden zusammen die Grammatik, die von dem Parser zur Konstruktion des abstrakten Strukturbaums benutzt wurde (siehe letzte Folge). Die Attributierungsregeln beschreiben, wie neue Attribute aus bekannten berechnet werden. Dabei sind die Attribute den Baumknoten zugeordnet, an denen sie hängen. In der zur ersten Regel *Assign ::= Ident ':=' Expr* gehörenden Attributierung steht *Ident.env* für das Umgebungsattribut des Bezeichners, dem durch die Anweisung ein Wert zugewiesen werden soll. Die erste Attributierungsregel *Ident.env := Assign.env* bedeutet also, daß das Umgebungsattribut des obersten Baumknotens aus Abb. 3 dem Umgebungsattribut des Bezeichners zugewiesen wird. Genauso beschreibt *Expr.env := Assign.env*, daß dasselbe Umgebungsattribut auch an den Wurzelknoten des arithmetischen Ausdrucks auf der rechten Seite des ':=' vererbt wird.

Die Funktion *LookUpType* dient dazu, aus der übergebenen Umgebung und einem Bezeichnernamen den Typ des Bezeichners zu ermitteln, und wird in der Regel von der Definitionstabelle zur Verfügung gestellt. In dem Beispiel wird der Einfachheit halber davon ausgegangen, daß es nur drei verschiedene Typen gibt, nämlich ganze Zahlen (*INTEGER*), Fließkommazahlen (*REAL*) und Arrays. Die Funktion wird in der dritten Attributierungsregel verwendet, um den *primode* des Bezeichners zu ermitteln, der das Ziel der Zuweisung ist. Die nächsten zwei Attributierungsregeln legen nur noch den *postmode* der beiden Nachfolgerknoten der Zuweisung fest und besagen, daß die Zuweisung erwartet, daß die beiden Typen mit dem *primode*, also dem aus der Definitionstabelle ermittelten Typ, des Bezeichners übereinstimmen. An dieser Stelle könnte man natürlich einwenden, daß die Zuweisung an *Ident.postmode* recht sinnlos ist. In dieser einfachen Grammatik ist sie auch überflüssig. Da sie die Attribute aber etwas einheitlicher gestaltet und in einer komplexeren Grammatik auch nötig ist, wurde sie hier aufgeführt.

Die Zusicherungen, die in jeder Regel durch das Schlüsselwort *CONDITION* gekennzeichnet sind, dienen der Überprüfung von semantischen Bedingungen. In unserem Beispiel muß in der Regel *Expr ::= Expr '[' Expr ']'* zum Beispiel sichergestellt werden, daß der Ausdruck, der zur Indizierung des Arrays verwendet wird, vom Typ *INTEGER* ist. Wird eine solche semantische Bedingung verletzt, so wird eine entsprechende Fehlermeldung ausgegeben.

Zusammenfassend kann man sagen, daß die semantische Analyse aus einer geschickten Attributierung des Strukturbaums besteht. Durch die Attributierung werden in den einzelnen Knoten des Baums zusätzliche Informationen gespeichert, die später bei der Codeerzeugung gebraucht werden. Außerdem wird das Programm auf semantische Fehler untersucht. Wie eine solche Attributierung programmiert wird und welche Probleme dabei auftreten können, werden wir uns im folgenden ansehen.

Der letzte Schritt

Da die attributierte Grammatik aus Listing 1 schon eine programmähnliche Form besitzt, liegt es nahe, eine Implementation der semantischen Analyse von ihr abzuleiten. Die Regeln des abstrakten Strukturbaums, die dort durch das Schlüsselwort *RULE* gekennzeichnet sind, müssen dazu in eine Typbeschreibung des Strukturbaums umgesetzt werden. Danach können die Regeln unter *ATTRIBUTATION* und die semantischen Bedingungen unter *CONDITION* in Prozeduren implementiert werden, die die notwendige Attributierung und die Tests durchführen. Eine Umsetzung in Modula-2 ist in Listing 2 abgebildet.

Die Regel *Assign ::= Ident ':=' Expr* wurde in den Typ *StatNode* umgesetzt. Die beiden Record-Felder *lhs* (left hand side) und *rhs* (right hand side) stehen dabei für die Teilbäume, die in der Regel *Ident* bzw. *Expr* heißen. Für eine komplette Programmiersprache, die sicherlich außer Zuweisungen noch andere Arten von Anweisungen kennt, muß *StatNode* natürlich noch um Alternativen für diese anderen Anweisungen erweitert werden; ähnlich wie dies im folgenden für arithmetische Ausdrücke beschrieben wird.

Da die drei restlichen Regeln auf der linken Seite des '::=' (bebecomes) alle dasselbe Nichtterminal, nämlich *Expr*, besitzen, werden sie zu einem Record namens *ExprNode* zusammengefaßt. Die Unterscheidung der drei möglichen Arten eines Expr-Knotens geschieht durch die Benutzung eines varianten Records. Die drei Varianten *plus*, *arrAc* (array access) und *ident* stehen dabei für die Regeln, die den Teilbaum für eine Summe, einen Array-Zugriff bzw. einen einzelnen Bezeichner beschreiben. Die ersten zwei bestehen dabei aus je zwei gleichartigen Unterbäumen. Der dritte enthält nur den Namen des Bezeichners, der hier vom Typ *Symbol* ist und sich als intrinsisches Attribut von Anfang an im Baum befindet, d.h. vom Parser eingetragen wird. Der Typ *Symbol* wird von der Bezeichnertabelle importiert.

Der Record *ExprNode* enthält unabhängig von der Variante noch die zwei zusätzlichen Felder *pri* und *post*, die zur Speicherung der Attribute *primode* bzw. *postmode* verwendet werden. Warum das Attribut *env* hier nicht auftaucht, werden wir später noch sehen.

Bleibt nur noch der Record *TypeNode* übrig. Er wird hier verwendet, um einen Typ zu repräsentieren. Dies ist entweder einer der beiden Grundtypen *INTEGER* oder *REAL* oder ein durch den Datentypkonstruktor (siehe Folge 2) Array aus zwei anderen Typen zusammengesetzter Typ.

Kletterpartie

Die auf die Typdeklaration folgenden Prozeduren implementieren die Attributierungsregeln und semantischen Bedingungen. Leider ist es dabei nicht möglich, einfach nacheinander die Berechnungen für die einzelnen Knoten des Baums durchzuführen. Wie wir weiter oben schon gesehen haben, hängen die einzelnen Attribute teilweise stark voneinander ab und müssen in der richtigen Reihenfolge berechnet werden. Wir hatten uns dort auch schon überlegt, daß die Berechnungen in maximal zwei Baumdurchläufen ausführbar sind.

Im ersten Durchlauf wird das Attribut *env* von oben nach unten vererbt und in den Blättern mittels *env* und *name* der a priori-Typ (*primode*) für die Bezeichner ermittelt. Außerdem wird das Attribut

primode der inneren Knoten aus deren Nachfolgerknoten berechnet.

Da das Attribut *env* nur verwendet wird, um zusammen mit dem Bezeichnernamen den Typ eines Bezeichners aus der Definitionstabelle auszulesen, ist es reine Speicherplatzverschwendung, wenn *env* in dem attributierten Strukturbaum gespeichert wird. Es wird daher im ersten Baumdurchlauf (Traversierung) als Parameter von einer Prozedur an die nächste weitergegeben. Die Prozedur *AttrAssign* implementiert die Attributierung der Regel *Assign ::= Ident ':=' Expr*. Sie wird mit dem bisher noch nicht attributierten abstrakten Strukturbaum und dem aktuellen Umgebungsattribut aufgerufen und berechnet mit der Funktion *LookUpType* der Definitionstabelle zuerst den Typ des Bezeichners, der auf der linken Seite des Zuweisungszeichens ':=' steht. Da der *postmode* von *Ident* und *Expr* nur von dem Typ des Bezeichners abhängt (*Ident.primode*), wird er in dieser Prozedur gleich mitberechnet und somit der zweite Durchlauf für den Zuweisungsknoten eingespart. Zu guter Letzt wird noch der erste (*AttrExpr1*) und dann der zweite Durchlauf (*AttrExpr2*) für die Attributierung des arithmetischen Ausdrucks, der auf der rechten Seite der Zuweisung steht, aufgerufen.

In *AttrExpr1* wird aufgrund der Art des zu bearbeitenden Baumknotens eine Fallunterscheidung durchgeführt, die in einen von drei möglichen Fällen verzweigt. Jeder Fall implementiert die Attributierungsregeln für den ersten Baumdurchlauf einer der drei möglichen Arten von Strukturbaumknoten eines arithmetischen Ausdrucks. In den ersten zwei Fällen wird die Prozedur *AttrExpr1* zuerst rekursiv für den jeweils linken und rechten Teilbaum des aktuellen Knotens aufgerufen und dabei das Attribut *env* als Prozedurparameter im Baum nach unten vererbt. In einem Blatt des Baumes angekommen, das einen Bezeichner darstellt, wird durch die Funktion *LookUpType* unter Zuhilfenahme des intrinsischen Attributs *name* und dem eben geerbten Attribut *env* der a priori-Typ (*primode*) des Bezeichners ermittelt.

Nach Beendigung des rekursiven Aufrufs wird in dem Baumknoten der jeweilige a priori-Typ (*primode*) unter Einbeziehung der Nachfolgerknoten berechnet. Bei einem Additionsknoten wird der Typ *integer* gewählt, wenn beide Summanden von Typ *integer* sind, sonst wird *real* genommen. In jedem Fall wird der entsprechende Knoten eines neuen Typbaums (*TypeTree*) durch den Aufruf einer entsprechenden Funktion erzeugt (*MakeRealType* bzw. *MakeIntType*). Im Fall eines Array-Zugriff-Knotens hängt der a priori-Typ nur von dem Typ des linken Teilbaums, also dem Array ab. Es ist nämlich der Elementtyp des Arrays, der mit der Hilfsfunktion *ElemType* ermittelt wird. Der a priori-Typ des rechten Teilbaums, also des Index', wird allerdings darauf überprüft, daß er vom Typ *integer* ist. Schließlich kann ein Array schlecht durch Fließkommazahlen indiziert werden.

Nachdem der erste Durchlauf komplett beendet ist, wird durch *AttrAssign* der zweite Durchlauf, also *AttrExpr2*, aufgerufen. Auch hier wird wieder anhand der Art des Baumknotens unterschieden. In den ersten beiden Fällen wird in jedem Knoten lediglich der a posteriori-Typ (*postmode*) der beiden Nachfolgerknoten berechnet und *AttrExpr2* dann rekursiv für diese Nachfolger aufgerufen. Der Test auf die Typverträglichkeit von *postmode* und *primode* ist bei allen drei Knotentypen gleichartig und wurde deshalb am Ende der Prozedur für alle drei Fälle zusammengefaßt. Was dieser Test genau macht, werden wir später noch sehen.

Durch diese Attributierung des Baums kann der Codegenerator sehr einfach entscheiden, ob er für eine Additionsoperation eine Fließkomma- oder eine Ganzzahladdition zu erzeugen hat. Er muß sich nur den a priori-Typ (*primode*) des Additionsknotens ansehen. Dort steht der Typ der Addition.

Zusammenlegung

Nachdem wir nun gesehen haben, wie man von einer attributierten Grammatik zu einem lauffähigen Programm gelangen kann, wollen wir uns noch kurz zwei Optimierungen ansehen.

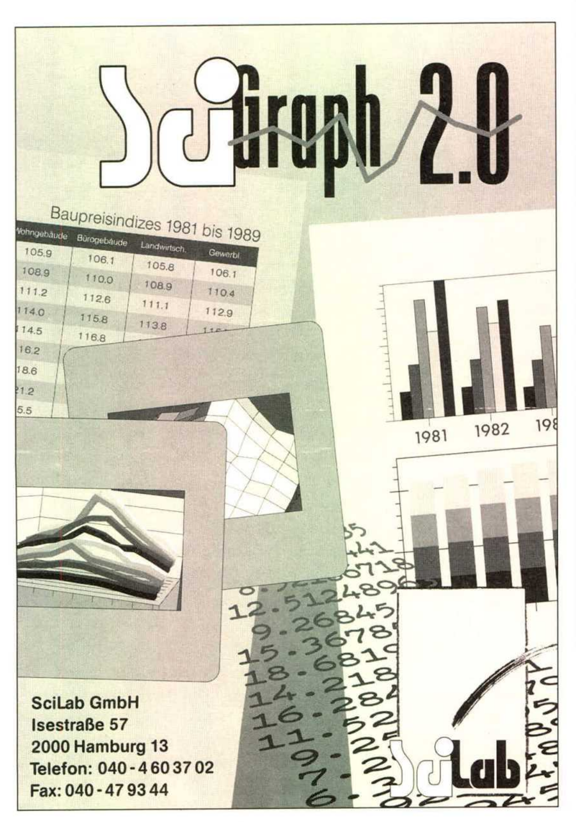
Betrachtet man sich das Zusammenspiel der beiden Durchläufe etwas genauer, so fällt auf, daß sich die beiden zu einem Durchlauf zusammenfassen lassen. Dies ist möglich, da der a posteriori-Typ eines Knotens nur vom a priori-Typ seines direkten Vorgängers (im Fall der Addition) oder seines linken Bruders (im Fall des Array-Zugriffs) abhängt und somit sofort nach der Rückkehr zur aufrufenden Rekursionsebene vom Aufrufer berechnet werden kann. Den erneuten rekursiven Aufruf spart man sich, indem der Test auf die Typverträglichkeit immer vom Vorgänger des zu testenden Knotens, statt von dem Knoten selber, ausgeführt wird. Ein Additionsknoten testet also die Typen seiner beiden Summanden, und die Zuweisung (*AttrAssign*) muß den Test für den Wurzelknoten des Ausdrucks auf der rechten Seite des Zuweisungszeichens (':=') machen.

Nachdem wir jetzt nur noch einen Baumdurchlauf haben, können wir noch einen Schritt weiter gehen und die semantische Analyse in den Parser integrieren und damit den expliziten Aufbau des Strukturbaums völlig vermeiden. Dazu erinnern wir uns noch einmal an die letzte Folge. Die Implementation eines Parsers nach der Methode des rekursiven Abstiegs war unserem jetzigen Baumdurchlauf sehr ähnlich. Für jedes Nichtterminal der Grammatik wurde eine Prozedur geschrieben, die wiederum zur Akzeptanz eines anderen Nichtterminals die zugehörige Prozedur aufruft. Im Gegensatz zur semantischen Analyse wurde die Entscheidung darüber, welche der möglichen Regeln im Einzelfall angewendet werden muß, durch das nächste zu akzeptierende Token gefällt. Diese Entscheidung war beim Parser aber auch ausschlaggebend für die Grammatikregel, für die sich der Parser entschieden hat, und damit für den Strukturbaumknoten, der erzeugt wurde. Aber genau an diesem Strukturbaumknoten orientiert sich die Regelauswahl der semantischen Analyse.

Es liegt also nahe, die Attributierungsregeln, die zu einer Grammatikregel gehören, in der entsprechenden Prozedur des rekursiven Abstiegs zu implementieren.

Diese Methode führt natürlich zu schön kompakten Compilern, die zudem recht wenig Speicherplatz verbauchen und meist auch schnell sind. Leider hat das Verfahren auch ein paar teilweise recht erhebliche Nachteile. Da der Strukturbaum nicht explizit aufgebaut wird, können die Attribute natürlich auch nicht in ihm gespeichert werden. Sie werden deshalb als lokale Variablen und Prozedurparameter der rekursiven Prozeduren realisiert. Sie sind somit für immer verloren, sobald die zugehörige Prozedur beendet wird. Dadurch muß natürlich auch die Codegenerierung für den entsprechenden Knoten durchgeführt werden, bevor die zugehörige Prozedur verlassen wird. Auf diese Art läßt sich leider nur Code bis zu einer bestimmten Codequalität erzeugen. Ein Hintertürchen bietet dabei die Möglichkeit, einen Zwischencode während des rekursiven Abstiegs zu erzeugen und die notwendigsten Attribute in der Definitionstabelle zu speichern. Der Zwischencode läßt sich nachträglich noch optimieren. Damit geht der Platzvorteil, den das Verfahren ursprünglich hatte, aber zum großen Teil wieder verloren.

Viel schwerer wiegt allerdings, daß die semantische Analyse der zu übersetzenden Programmiersprache überhaupt in einem Durchlauf durch den Strukturbaum ausführbar sein muß. Schon für unser kleines Beispiel waren die Abhängigkeiten der einzelnen Attribute recht verworren. In einer kompletten Programmiersprache sind


SciGraph 2.0
Baupreisindizes 1981 bis 1989
1981
1982
SciLab

die Verhältnisse teilweise noch sehr viel komplizierter und lassen sich manchmal nur mit mehreren Durchläufen beherrschen. Und selbst das reicht nicht, denn beim rekursiven Abstieg wird der Baum immer linksrekursiv durchlaufen. Das bedeutet, daß bei mehr als einem Nachfolger eines Strukturbaumknotens immer zuerst der in der Grammatik am weitesten links stehende durchlaufen wird. In einer Modula-2-Variablendeklaration der Form *VAR i, j, k, l: INTEGER* hängt der Typ der Bezeichner *i*, *j*, *k* und *l* aber von einem Teilbaum ab, der weiter rechts steht, nämlich von dem Typ rechts des Doppelpunkts. In diesem Fall muß man alle Bezeichner (und das können beliebig viele sein) zwischenspeichern und ihnen später den Typ zuweisen. Sobald der Parser nicht nach dem Prinzip des rekursiven Abstiegs oder noch nicht einmal als LL-Parser realisiert ist, fällt diese Methode sowieso flach. Es gibt dann zwar auch noch Möglichkeiten, Parser und semantische Analyse zu integrieren, doch sind diese in der Regel aufwendiger als das gerade besprochene Verfahren.

Typzwang

Wie versprochen, kommen wir jetzt noch einmal kurz auf die mysteriösen Attribute *primode* und *postmode* sowie das Prädikat *Coercible* zurück. Die meisten Programmiersprachen kennen implizite Typkonversionen, das heißt, Werte, die von sich aus einen anderen Typ besitzen, als es von der Umgebung verlangt wird, werden automatisch in den richtigen Typ verwandelt. Dies setzt natürlich voraus, daß der ursprüngliche Typ (a priori-Typ) in den erwarteten (a posteriori-Typ) umgewandelt werden kann.

In unserer Beispielsprache ist es etwa möglich, eine Zuweisung wie *a := 2 + 3.1* hinzuschreiben. Der a priori-Typ der Zahl 2 ist Integer. Da eine Addition aber entweder zwei ganze oder zwei Fließkommazahlen summiert und eine Wandlung der Fließkommazahl einen Informationsverlust bedeutet würde, ist der von dem Additionsoperator erwartete Typ (a posteriori-Typ) der Zahl 2 Real.

Das Prädikat *Coercible* testet, ob die beiden übergebenen Typen ineinander überführt werden können. Da eine ganze in eine Fließkommazahl gewandelt werden kann, ist es in der obigen Addition erfüllt.

Auch in einer Programmiersprache wie Modula-2, die nahezu keine für den Programmierer direkt sichtbaren impliziten Typwandlungen erlaubt, sind solche Typzwänge wichtig. Wird ein Prozedurparameter einmal *call by value* und ein anderes Mal *call by reference* (VAR-Parameter) übergeben, so wird der Parameter innerhalb der Prozedur doch immer auf dieselbe Art verwendet. Auch ein als *call by reference* übergebener Parameter muß bei seiner Benutzung nicht explizit dereferenziert werden. Dies geschieht implizit und stellt damit auch eine Form der impliziten Typwandlung dar. Ähnliches gilt für linke und rechte Seiten von Zuweisungen. Diese Problematik wurde in unserem Beispiel völlig ignoriert, da sie es aufgebläht hätte, ohne wesentlich neue Einsichten zu vermitteln.

Zu kurz ist in dieser Beschreibung sicherlich die Betrachtung der Namensanalyse gekommen. Die Hauptarbeit der Namensanalyse liegt aber bei der Definitionstabelle und weniger bei den Attributierungsregeln, die die Funktionen der Definitionstabelle benutzen. In der ersten Folge wurde schon erwähnt, daß sich in [3] eine sehr elegante Lösung für Sprachen, die beliebig verschachtelte lokale Deklarationen zulassen, befindet.

Übergang

Nachdem das zu übersetzende Programm jetzt vollständig analysiert ist, wird es langsam Zeit, die zweite große Aufgabe eines Compilers in Angriff zu nehmen. Aus dem attributierten Strukturbaum muß ein Maschinenprogramm erzeugt werden, daß in seinem Verhalten dem Quellprogramm entspricht. Es gibt allerdings einige Gründe, dies nicht in einem Schritt zu machen, sondern in Teilaufgaben zu zerlegen. Gründe dafür sind, daß der Schritt bei einer sehr komplizierten Quellsprache unter Umständen zu groß ist oder das Programm vorher noch optimiert werden soll. Außerdem kann eine sorgfältig ausgewählte Zwischensprache das Umschreiben des Compilers auf eine neue Zielsprache stark erleichtern. Da die Maschinensprache des Zielprozessors oft sehr viele Ausnahmen besitzt, was die möglichen Adressierungsarten und ähnliches angeht, kann eine Zwischenebene, die dazu dient, diese Unebenheiten auszubügeln, den Compiler übersichtlicher, kürzer und schneller machen.

Statt den attributierten Strukturbaum sofort in die Zielsprache umzuwandeln, wird er deshalb oft erst einmal in einen Zwischencode transformiert. Der Zwischencode kann natürlich von Compiler zu Compiler und abhängig von der Zielsprache sehr verschieden sein. In der Regel wird ein Code gewählt, der in seiner Struktur der Zielsprache schon recht ähnlich ist. Wie weit er von der Zielsprache entfernt ist, hängt in erster Line von den Aufgaben ab, die ihm zugedacht sind. Möchte man lediglich die gröbsten Unebenheiten in der zu erzeugenden Maschinensprache ausbügeln, wird man als Zwischencode einfach die Zielsprache, erweitert um ein paar Adressierungsmöglichkeiten und eventuell auch Befehle, benutzen. Dieser Fall soll uns jetzt nicht so sehr interessieren, er gehört thematisch eher in die nächste Folge.

Sehr viel interessanter ist eine Ebene, die zusätzliche Optimierungen erlaubt und eine Umstellung des Compilers auf eine andere Zielsprache erleichtert. Für diese Aufgabe wird meist eine Zwischensprache gewählt, die in ihrer Struktur einer konventionellen Maschinensprache möglichst ähnlich, aber noch nicht auf einen speziellen Prozessor festgelegt ist. Außerdem ist es für manche Optimierungen wichtig, einige Hochsprachenkonstrukte, wie zum Beispiel Array-Zugriffe, noch nicht in Einzeloperationen aufzulösen.

Diese Anforderungen werden zum Beispiel von einem 3-Adreßcode erfüllt, der die folgenden Befehlsarten enthält. Ein Ausdruck wie *x := y <Op> z* oder *x := <Op> (x, y)* erlaubt das Verknüpfen zweier Operanden durch eine Operation (<Op>) wie Addition, logisches Und usw. Entsprechendes gilt für einstellige Operatoren nach dem Schema *x := <Op> y*. Zum Kopieren von Werten werden Anweisungen der Form *x := y* verwendet. Ein Array-Zugriff erfolgt etwa durch *x := a[y]*. Außerdem gibt es noch Anweisungen wie *call* und *return*, um Prozedur- und Funktionsaufrufe zu implementieren, sowie bedingte und unbedingte Sprungbefehle, um die Verzweigung des Programmflusses zu realisieren.

Attributierungsregeln, die einen attributierten Strukturbaum unserer Beispielgrammatik in Zwischencode umwandeln, enthält Listing 3. Die Anweisungen der Zielsprache sind der Übersichtlichkeit halber in geschweiften Klammern und im Klartext angegeben. Der Operator '++' dient dem Zusammenfügen zweier Sequenzen von 3-Adreßcodeanweisungen. Die Funktion 'GenerateNewTemp' erzeugt bei jedem Aufruf eine neue Zwischenvariable, z.B. die Variablen *t1*, *t2* usw. Die Information der semantischen Analyse über den Typ einer Addition, also Ganzzahl- oder Fließkommaaddition, wird in der IF-Anweisung der zweiten Regel ausgenutzt.

Die Grammatik in Listing 3 beinhaltet allerdings noch einen Fehler. Falls der *Expr[0].primode* und *Expr[0].postmode* in einer der drei letzten Regeln nicht übereinstimmen, muß noch eine automatische Typwandlung durchgeführt werden, bevor das Ergebnis *Expr[0].Loc* zugewiesen wird. Aus Gründen der Übersicht wurden

die entsprechenden Anweisungen weggelassen.

Wenn wir uns Abb. 1 noch einmal anschauen, sehen wir, daß die beiden attributierten Grammatiken (Listing 1 und Listing 3) genau das leisten, was wir uns am Anfang erhofft haben. Der Strukturbaum wird in den gewünschten Zwischencode übersetzt, und nebenbei hat die semantische Analyse noch eventuell vorhandene Typfehler aufgedeckt.

Generatoren

Generatoren zur automatischen Erzeugung der semantischen Analyse sind sehr viel komplizierter als etwa Parser- oder Scanner-Generatoren. Dies liegt im wesentlichen daran, daß die Aufgaben der semantischen Analyse vielfältiger als die der anderen Phasen des Compilers sind. Die Art und Komplexität der Aufgabe hängt sehr stark von der zu übersetzenden Programmiersprache ab und variiert deshalb stark von Compiler zu Compiler. Aus diesem Grund gibt es auch keine Standardwerkzeuge wie REX oder YACC. Trotzdem waren die Compiler-Bauer in den letzten Jahren nicht untätig und haben sich einige sehr nützliche Programme ausgedacht.

Prinzipiell gibt es zwei verschiedene Ansätze, um die Programmierung der semantischen Analyse zu vereinfachen. Erstens gibt es Programme, die in der Lage sind, eine Eingabe in Form einer attributierten Grammatik in ein lauffähiges Programm zu übersetzen. Die Attributierungsregeln werden dabei normalerweise in einer gängigen Programmiersprache wie Modula-2 oder C angegeben. Diese Methode entspricht in ihrem Konzept dem Vorgehen von Parser-oder Scanner-Generatoren. Zusätzlich werden dabei oft Generatoren zur automatischen Generierung von Modulen zur Verwaltung des Strukturbaums und ähnlichem eingesetzt.

Die zweite Möglichkeit ist die Entwicklung von speziellen Programmiersprachen, die es erlauben, die Regeln der semantischen Analyse sehr viel übersichtlicher zu formulieren, als dies in einer der gängigen Programmiersprachen der Fall ist.

Dabei stellt sich natürlich die Frage, wie bei den beiden Möglichkeiten die Auswertungsreihenfolge der einzelnen Attribute festgelegt wird und wieviel Arbeit der Computer dem Programmierer eines Compilers jeweils abnimmt. Im ersten Fall wird die Auswertungsreihenfolge der Attribute durch den Generator festgelegt. Dies geschieht etwa durch geordnete attributierte Grammatiken (näheres zum Beispiel in [3], aber Achtung: schwerverdaulich). Da das Finden einer möglichst optimalen Auswertungsreihenfolge der Attribute eine sehr komplizierte Arbeit ist, ist ein solcher Generator ein sehr nützliches Werkzeug. Außerdem ist es sehr viel einfacher, die Auswertung der Attribute zu ändern, wie dies etwa für eine Erweiterung des Compilers nötig sein kann. Die zweite Möglichkeit überläßt es dem Programmierer, die Reihenfolge der Auswertung festzulegen. Allerdings läßt sich durch eine Integration des Aufbaus des Strukturbaums und der Regelauswahl eine sehr übersichtliche Darstellung der semantischen Analyse erreichen. Das Compiler-Bauwerkzeug GENTLE ([2]) erreicht das zum Beispiel durch eine deklarative Beschreibung des Compilers, die der von Prolog-Programmen ähnelt. Da der Parser auch in diese Beschreibung integriert ist, können komplette Compiler in GENTLE geschrieben werden. Zum Erzeugen des Parsers greift das System übrigens auf den Parser-Generator LALR zurück, für den automatisch eine Eingabedatei erzeugt wird. Nur die Scanner-Beschreibung muß von Hand für den Scanner-Generator REX geschrieben werden. Für die Beschreibung des Strukturbaums und für eventuelle Zwischensprachen wird ein System algebraischer Datentypen benutzt, daß eine für den Programmierer sehr übersichtliche Schreibweise und damit Programmierung zuläßt.

```
 1:  RULE Assign ::= Ident ':=' Expr.
 2:  ATTRIBUTION
 3:    Ident.env := Assign.env;
 4:    Expr.env := Assign.env;
 5:    Ident.primode := LookUpType (Ident.env,
                                     Ident.name);
 6:    Ident.postmode := Ident.primode;
 7:    Expr.postmode := Ident.primode;
 8:
 9:  RULE Expr ::= Expr '+' Expr.
10:  ATTRIBUTION
11:    Expr[1].env := Expr[0].env;
12:    Expr[2].env := Expr[0].env;
13:    Expr[0].primode := IF Expr[1].primode = real
                          OR Expr[2].primode = real
14:                       THEN real ELSE integer END;
15:    Expr[1].postmode := Expr[0].primode;
16:    Expr[2].postmode := Expr[0].primode;
17:  CONDITION
18:    Coercible (Expr[0].primode, Expr[0].postmode);
19:
20:  RULE Expr ::= Expr '[' Expr ']'.
21:  ATTRIBUTION
22:    Expr[1].env := Expr[0].env;
23:    Expr[2].env := Expr[0].env;
24:    Expr[0].primode := ElemType (Expr[1].primode);
25:    Expr[1].postmode := Expr[1].primode;
26:    Expr[2].postmode := IndexType
                           (Expr[1].primode);
27:  CONDITION
28:    IsArrayType (Expr[1].primode);
29:    Expr[2].primode = integer;
30:    Coercible (Expr[0].primode, Expr[0].postmode);
31:
32:  RULE Expr ::= Ident.
33:  ATTRIBUTION
34:    Expr.primode := LookUpType (Expr.env,
                                   Ident.name);
35:  CONDITION
36:    Coercible (Expr.primode, Expr.postmode);
```

Listing 1: Attributierte Grammatik für das Zuweisungsbeispiel

```
 1:  TYPE    ExprKind = (plus, arrAc, ident);
 2:          TypeKind = (integer, real, array);
 3:
 4:          StatTree = POINTER TO StatNode;
 5:          StatNode = RECORD       (*  Assignment  *)
 6:                       lhs: ExprTree;
 7:                       rhs: ExprTree;
 8:                     END;
 9:
10:          ExprTree = POINTER TO ExprNode;
11:          ExprNode = RECORD       (*  Expression  *)
12:                       CASE kind: ExprKind OF
13:                         plus  : lhs,
14:                                 rhs  : AttrTree|
15:                         arrAc : arr,
16:                                 idx  : AttrTree|
17:                         ident : name: Symbol|
18:                       END;
19:                       pri : TypeTree;
20:                       post: TypeTree;
21:                     END;
22:
23:          TypeTree = POINTER TO TypeNode;
24:          TypeNode = RECORD       (*  Typausdruck  *)
25:                       CASE kind: TypeKind OF
26:                         integer: |
27:                         real   : |
28:                         array  : elemType:
                                     TypeTree;
29:                                  idxType :
                                     TypeTree|
30:                       END;
31:                     END;
32:
33:
34:  (*  Attributiere eine Zuweisung
35:   *)
36:  PROCEDURE AttrAssign (tree: StatTree; env: Env);
37:    BEGIN
38:      WITH tree^
39:      DO                                         →
```

```
 40:       lhs^.pri := LookUpType (env, lhs^.name);
 41:       lhs^.post := lhs^.pri;
 42:       rhs^.post := lhs^.pri;
 43:       AttrExpr1 (rhs, env);
 44:       AttrExpr2 (rhs);
 45:     END;
 46:   END AttrAssign;
 47:
 48:  (* Attributiere eine Ausdruck: Erster Durchlauf
 49:   *)
 50:  PROCEDURE AttrExpr1 (tree: ExprTree; env: Env);
 51:   BEGIN
 52:     WITH tree^ DO
 53:      CASE kind OF
 54:
 55:       plus : AttrExpr1 (lhs, env);
 56:              AttrExpr1 (rhs, env);
 57:              IF lhs^.pri^.kind = real
 58:                 OR rhs^.pri^.kind = real
 59:              THEN pri := MakeRealType ();
 60:              ELSE pri := MakeIntType () END|
 61:
 62:       arrAc: AttrExpr1 (arr, env);
 63:              AttrExpr1 (idx, env);
 64:              pri := ElemType (arr^.pri);
 65:              IF idx^.pri^.kind # integer
 66:                 OR arr^.pri^.kind # array
 67:              THEN RaiseError END|
 68:
 69:       ident: pri := LookUpType (env, name)|
 70:
 71:      END;
 72:     END;
 73:   END AttrExpr1;
 74:
 75:  (* Attributiere eine Ausdruck: Zweiter Durchlauf
 76:   *)
 77:  PROCEDURE AttrExpr2 (tree: ExprTree);
 78:   BEGIN
 79:     WITH tree^ DO
 80:      CASE kind OF
 81:
 82:       plus : lhs^.post := pri;
 83:              rhs^.post := pri;
 84:              AttrExpr2 (lhs);
 85:              AttrExpr2 (rhs)|
 86:
 87:       arrAc: arr^.post := arr^.pri;
 88:              idx^.post := IndexType (arr^.pri);
 89:              AttrExpr2 (arr);
 90:              AttrExpr2 (idx)|
 91:
 92:       ident: |
 93:
 94:      END;
 95:     IF NOT Coercible (pri, post) THEN RaiseError
                                          END
 96:     END;
 97:   END AttrExpr2;
 98:
 99:  (* Liefert den Elementtyp eines Arrays.
100:   *)
101:  PROCEDURE ElemType (type: TypeTree): TypeTree;
102:   BEGIN
103:     IF type^.kind # array THEN RaiseError
104:     ELSE
105:      RETURN type^.elemType
106:     END;
107:   END ElemType;
108:
109:  (* Liefert den Indextyp eines Arrays.
110:   *)
111:  PROCEDURE IndexType (type: TypeTree): TypeTree;
112:   BEGIN
113:     IF type^.kind # array THEN RaiseError
114:     ELSE
115:      RETURN type^.idxType
116:     END;
117:   END ElemType;
```

Listing 2: Routinen zur Attributierung des Zuweisungsbeispiels

GENTLE übersetzt die Beschreibung in ein C-Programm, das den beschriebenen Compiler implementiert.

Die Masse der Denkarbeit beim Schreiben der semantischen Analyse liegt allerdings in dem Finden einer für die zu übersetzende Programmiersprache geeigneten attributierten Grammatik. Diese Arbeit muß nach wie vor der Compiler-Bauer erledigen, und sie wird ihm wohl auch nicht so schnell von einem Computer abgenommen. Doch kann dieser sehr viel Routinearbeit übernehmen und den Programmierer dadurch entlasten.

Eine interessante Enwicklung ist auch die, daß in die Definition einer Programmiersprache eine Beschreibung ihrer Semantik aufgenommen wird. Sobald aus dieser Beschreibung mehr oder weniger automatisch eine Attributierung und eventuell Codeerzeugung abgeleitet werden kann, ist es wesentlich leichter, einen Standard für diese Sprache zu schaffen. Da dies zu einer leichteren Portierung von Programmen führt, ist diese Entwicklung für alle Software-Entwickler von Vorteil.

Wie geht's weiter?

Langsam nähern wir uns in der Übersetzung der Maschine und damit auch dem Ende der Serie über Compiler-Bau. In der nächsten und zugleich letzten Folge werden wir uns intensiv mit der Codeerzeugung auseinandersetzen. Dabei wird uns in erster Linie interessieren, mit welchen Mitteln man den erzeugten Code verbessern kann.

Manuel Chakravarty

Literatur:

[1] Aho/Sethi/Ullman: „Compilers: Principles, Techniques and Tools", Addison-Wesley

[2] Vollmer: „The Compiler Construction System GENTLE", bisher erst internes Papier der GMD Karlsruhe

[3] Waite/Goos: „Compiler Construction", Springer


```
 1:  RULE Assign ::= Ident ':=' Expr.
 2:  ATTRIBUTION
 3:    Assign.code := Expr.code ++ {Ident.name :=
                                     Expr.loc;};
 4:
 5:  RULE Expr ::= Expr '+' Expr.
 6:  ATTRIBUTION
 7:    Expr[0].loc := GenerateNewTemp ();
 8:    Expr[0].code := Expr[1].code ++ Expr[2].code
 9:                    ++ IF Expr[0].primode =
                        integer
10:                        THEN {Expr[0].loc :=
                               add_int
                               (Expr[1].loc,
                               Expr[2].loc);}
11:                        ELSE {Expr[0].loc :=
                               add_real
                               (Expr[1].loc,
                               Expr[2].loc);}
12:                        END;
13:
14:  RULE Expr ::= Expr '[' Expr ']'.
15:  ATTRIBUTION
16:    Expr[0].loc := GenerateNewTemp ();
17:    Expr[0].code := Expr[1].code ++ Expr[2].code
18:                    ++ {Expr[0].loc :=
                       Expr[1].loc[Expr[2].loc];};
19:
20:  RULE Expr ::= Ident.
21:  ATTRIBUTION
22:    Expr.loc := Ident.name;
23:    Expr.code := {};
```

Listing 3: Zwischencodeerzeugung für die Beispielgrammatik

Datenstrukturen & Algorithmen

in Omikron.BASIC und Modula 2

Teil 4: String-Sortieren mit Radixsort

In den vorangegangenen drei Kursteilen haben wir die theoretischen Grundlagen zum Thema Datenstrukturen und zum Radixsort vorgestellt. Wir wenden wir uns nun der Praxis zu und sortieren Zeichenketten. Die Algorithmen entwickeln wir in Omikron.BASIC-Syntax und stellen anschließend die Modula 2-Lösung vor.

Radixsort basiert auf der wiederholten Anwendung des Behältersortierens (Binsort). Zuerst verteilt man bezüglich des am wenigsten signifikanten Schlüssels in die Behälter, zum Schluß nach dem mit der höchsten Priorität. Die zugrundeliegende Datenstruktur ist eine lineare Liste. Diese interpretieren wir als Schlange. Bild 16 aus dem dritten Kursteil zeigt den allgemeinen Radixsort-Algorithmus. Wie wenden wir diesen Algorithmus auf das Sortieren von Strings an? Die meisten Sortieralgorithmen betrachten gesamte Zeichenketten als Schlüssel. Auch der Quicksort-Algorithmus aus Omikron.BASIC geht diesen Weg. Bei Radixsort splitten wir die Zeichenketten in ihre Komponenten, die Zeichen. Das erste Zeichen ist am meisten und das letzte am wenigsten signifikant. Jeder Durchlauf verteilt bezüglich einer Zeichenposition, beginnend mit dem letzten Zeichen. Als Beispiel betrachten wir eine Schlange mit den folgenden sechs 'Wörtern':

CAC, DCA, ADC, BCD, BBA, DAB

Das zugrundeliegende Alphabet enthalte nur die vier Buchstaben 'A', 'B', 'C' und 'D'. Wir benötigen vier Behälter, die wir B(A), B(B), B(C) und B(D) nennen. Im ersten Durchlauf sortieren wir bezüglich des dritten Zeichens in die Behälter:

B(A): DCA, BBA
B(B): DAB
B(C): CAC, ADC
B(D): BCD

Nach dem Aufsammeln der Behälter lautet die Schlange

DCA, BBA, DAB, CAC, ADC, BCD.

Diese verteilen wir bezüglich des zweiten Zeichens in dieselben Behälter:

B(A): DAB, CAC
B(B): BBA
B(C): DCA, BCD
B(D): ADC

Wir erhalten die Schlange

DAB, CAC, BBA, DCA, BCD, ADC.

Abschließend sortieren wir diese bezüglich des Anfangsbuchstabens:

B(A): ADC
B(B): BBA, BCD
B(C): CAC
B(D): DAB, DCA

Nach dem Auflesen der Behälter ist die Schlange sortiert:

ADC, BBA, BCD, CAC, DAB, DCA

Alle Schlüsselwerte sind vom selben Datentyp: CHAR. Wir verteilen deshalb in allen Durchläufen in die selben Behälter. Der allgemeine Radixsort-Algorithmus aus Bild 16 dagegen beansprucht für jeden Schlüsselwert-Datentyp eigene Behälter.

Implementierung in Omikron.BASIC

Radixsort ordnet Daten, die als Schlange organisiert sind. In Omikron.BASIC realisieren wir Schlangen in Feldern. Die Reihenfolge der Elemente ergibt sich aus der Verzeigerung, nicht aus der Feldposition. Radixsort modifiziert ausschließlich die Zeiger. Die Daten werden nicht umgespeichert. Nach dem Sortieren enthält das Feld die Daten in derselben Reihenfolge wie vorher. Die sortierte Reihenfolge ergibt sich allein aus der geänderten Verzeigerung. Oft will man ein Feld sortieren. Die sortierte Reihenfolge soll sich aus der Feldposition ergeben und nicht aus einer zusätzlichen Verzeigerung. In diesem Fall erhalten wir folgende Grobstruktur:

1. Feld zu Schlange erweitern
2. Schlange sortieren
3. sortierte Schlange in Feld übertragen

Wir betrachten exemplarisch unsere Datenstruktur *Wl*: Das Array *Wl_Wort$()* speichert die zu sortierenden Wörter. *Wl_Next%()* enthält die Cursor auf die Nachfolger. *Wl* haben wir ausführlich in den ersten beiden Kursteilen beschrieben.

Schritt 1: Feld zu Schlange erweitern
Das Feld *Wl_Wort$()* enthalte *N* zu sortierende Strings in den Feldelementen eins bis *N*. Diese seien noch nicht als Liste verkettet. Das nullte Feldelement muß frei sein, da es nicht mit einem Cursor adres-

sierbar ist. Falls es ein Wort enthält, kopieren Sie dies an das Feldende. Für eine Schlange benötigen wir Cursor. Wir dimensionieren ein zusätzliches Feld *Wl_Next%()*. Die Liste generieren wir, indem wir die Wörter mittels *Wl_Next%()* verzeigern: Alle Cursor zeigen auf das Feldelement mit dem nächsthöheren Index. Das Listenende bekommt Null (NIL) zugewiesen. *Wl_Wort$()* bleibt bei diesen Operationen unberührt.

```
' Feld zu Schlange erweitern
FOR I%=1 TO N%-1
   Wl_Next%(I%)=I%+1
NEXT I%
Wl_Next%(N%)=0
```

Den Schlangenkopf und das Schlangenende speichern wir mit

```
S_Anfang%=1: S_Ende%=N%
```

Bild 17 verdeutlicht die Wirkungsweise.

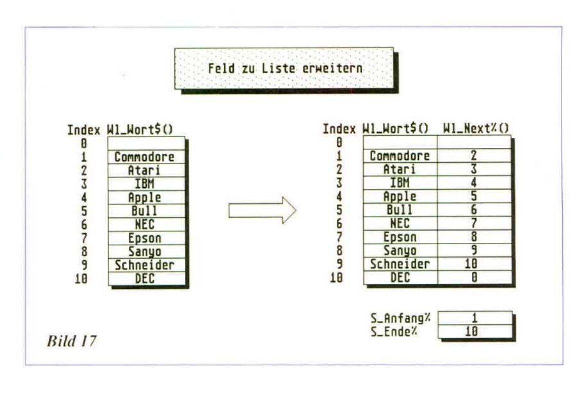


Bild 17

Schritt 2: Schlange sortieren

Im dritten Kursteil haben wir angeschnitten, wie wir das Behältersortieren in Omikron.BASIC implementieren. Diese Kenntnisse setzen wir voraus. Die Grobstruktur für Radixsort lautet:

```
FOR Char_Nr%=Max_Laenge% TO 1 STEP -1
   Binsort(Char_Nr%,S_Anfang%,S_Ende%)
NEXT Char_Nr%
```

Die Konstante *Max_Laenge%* legt die maximale String-Länge fest. Die Variable *Char_Nr%* kennzeichnet, welches Zeichen des Strings Schlüssel für die Sortierung ist. In der Prozedur *Binsort* verteilen wir wie folgt in die Behälter:

```
'S in Behälter sortieren
'Schlüssel ist das Char_Nr%-te Zeichen
WHILE S_Anfang%<>0 P%=Wl_Next%(S_Anfang%)
   ' Cursor
   'auf das zweite Listenelement
   'retten
   Wort$=Wl_Wort$(S_Anfang%)
   Ch%= ASC( MID$(Wort$,Char_Nr%,1))
   Ll_Anhaengen(B_Anfang%(Ch%),B_Ende%(Ch%),S_Anfang%)
   S_Anfang%=P%
WEND
```

Bitte beachten Sie die Änderung bei der Zuweisung an *Ch%*. Im dritten Kursteil haben wir nur das erste Zeichen als Schlüssel angesehen - hier ist es das *Char_Nr%*-te. Die Behälter adressieren wir mit dem ASCII-Wert der zu verteilenden Zeichen. *B(A)* aus dem obigen Beispiel entspricht somit der Schlange, die wir in *B_Anfang%(65)* und *B_Ende%(65)* speichern. Bild 18 demonstriert die Wirkungsweise des Algorithmus.

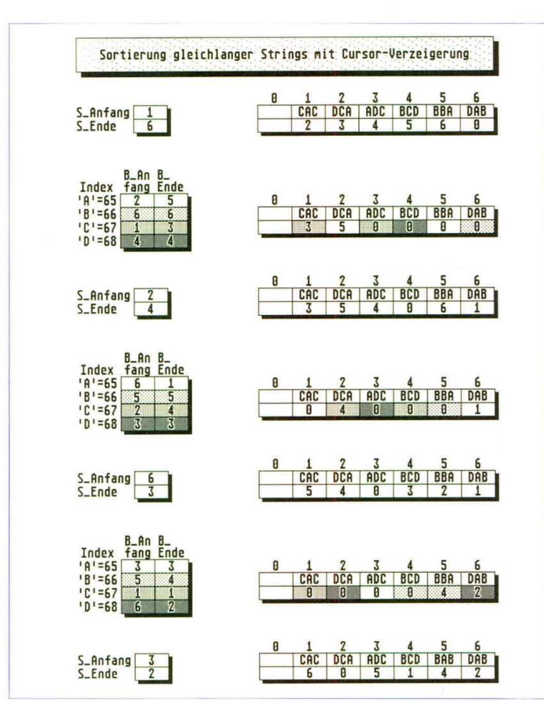


Bild 18

Verschieden lange Wörter

Bislang haben wir stillschweigend gleichlange Wörter sortiert. Wir sind damit analog zur konstanten Schlüsselfeldanzahl im Datentyp *ObjTyp* des dritten Kursteils vorgegangen. In der Praxis variiert die String-Länge zwischen Null und *Max_Laenge%*. Somit ist auch die Anzahl der Schlüssel variabel. Das BASIC quittiert uns den Zugriff auf ein zu kurzes Wort mit einem Programmabbruch. Wir testen daher bei jedem Wort, ob es für einen Sortierlauf lang genug ist. Wir verteilen erst, wenn *Char_Nr%* kleiner oder gleich der Wortlänge ist. Alle Wörter, die für einen Durchlauf zu kurz sind, sammeln wir in einem zusätzlichen Behälter. In Omikron.BASIC haben wir *B(0)* gewählt, da alle Behälter unterhalb von *Min_Ascii%* unbenutzt sind. *B(0)* ist das nullte Element des Behälterfeldes. Beim Sammeln der Behälter lesen wir zuerst den zusätzlichen Behälter auf. Anschließend sammeln wir wie gewohnt. Diese Reihenfolge ergibt sich aus der Tatsache, daß kurze Wörter in der alphabetischen Ordnung vor langen Wörtern mit gleichem Anfang stehen. Also, 'Maus' kommt vor 'Mausefalle'. Betrachten wir z.B. die folgende Schlange mit acht 'Wörtern':

CAB, BCD, DA, BADC, C, ACBD, AB, BD

Im ersten Durchlauf betrachen wir das vierte Zeichen. Alle Wörter, die kürzer als vier Zeichen sind, kommen nach *B(0)*. Die restlichen verteilen wir in die Behälter *B(A)*, *B(B)*, *B(C)* und *B(D)*.

B(0): CAB, BCD, DA, C, AB, BD
B(A): leer
B(B): leer
B(C): BADC
B(D): ACBD

Nach dem Auflösen der Behälter erhalten wir die Schlange

CAB, BCD, DA, C, AB, BD, BADC, ACBD.

Im zweiten Durchlauf verteilen wir bezüglich des dritten Zeichens.

B(0): DA, C, AB, BD
B(A): leer
B(B): CAB, ACBD
B(C): leer
B(D): BCD, BADC

Im dritten Durchlauf ordnen wir die Schlange

DA, C, AB, BD, CAB, ACBD, BCD, BADC

bezüglich des zweiten Zeichens:

B(0): C
B(A): DA, CAB, BADC
B(B): AB
B(C): ACBD, BCD
B(D): BD

```
Zeichenketten sortieren - Version 1

Vor der Sortierung:
Bruno  Anna  Klaus  Fritz  Kunigunde  Ira  Franz  Susanne  Bernd  Emil

Sortierläufe:
15   Bruno  Anna  Klaus  Fritz  Kunigunde  Ira  Franz  Susanne  Bernd  Emil
14   Bruno  Anna  Klaus  Fritz  Kunigunde  Ira  Franz  Susanne  Bernd  Emil
13   Bruno  Anna  Klaus  Fritz  Kunigunde  Ira  Franz  Susanne  Bernd  Emil
12   Bruno  Anna  Klaus  Fritz  Kunigunde  Ira  Franz  Susanne  Bernd  Emil
11   Bruno  Anna  Klaus  Fritz  Kunigunde  Ira  Franz  Susanne  Bernd  Emil
10   Bruno  Anna  Klaus  Fritz  Kunigunde  Ira  Franz  Susanne  Bernd  Emil
9   Bruno  Anna  Klaus  Fritz  Ira  Franz  Susanne  Bernd  Emil  Kunigunde
8   Bruno  Anna  Klaus  Fritz  Ira  Franz  Susanne  Bernd  Emil  Kunigunde
7   Bruno  Anna  Klaus  Fritz  Ira  Franz  Bernd  Emil  Susanne  Kunigunde
6   Bruno  Anna  Klaus  Fritz  Ira  Franz  Bernd  Emil  Susanne  Kunigunde
5   Anna  Ira  Emil  Bernd  Kunigunde  Susanne  Bruno  Klaus  Fritz  Franz
4   Ira  Anna  Susanne  Kunigunde  Emil  Bernd  Bruno  Franz  Fritz  Klaus
3   Ira  Franz  Klaus  Emil  Fritz  Anna  Kunigunde  Bernd  Susanne  Bruno
2   Bernd  Klaus  Emil  Anna  Ira  Franz  Fritz  Bruno  Kunigunde  Susanne
1   Anna  Bernd  Bruno  Emil  Franz  Fritz  Ira  Klaus  Kunigunde  Susanne

Nach der Sortierung:
Anna  Bernd  Bruno  Emil  Franz  Fritz  Ira  Klaus  Kunigunde  Susanne
```

Bild 19

Nach Auflesen der Behälter erhalten wir:

C, DA, CAB, BADC, AB, ACBD, BCD, BD

Diese Schlange verteilen wir abschließend bezüglich des ersten Zeichens:

B(A): AB, ACBD
B(B): BADC, BCD, BD
B(C): C, CAB
B(D): DA

Die sortierte Schlange lautet somit:

AB, ACBD, BADC, BCD, BD, C, CAB, DA

Kurze Wörter schieben wir solange in *B(0)* auf die Wartebank, bis wir bezüglich deren Länge verteilen. Listing 6 enthält die wesentlichen Programmteile zur Implementierung des Algorithmus in Omikron.-BASIC. Bild 19 zeigt die einzelnen Sortierläufe anhand von Beispieldaten. *Max_Laenge%* ist gleich 15. Das längste Wort, "Kunigunde", hat neun Buchstaben. Der Algorithmus führt somit sechs 'Leerläufe' am Anfang aus. In diesen sortieren wir nicht, sondern speichern die Daten aus der Schlange *S* sukzessive in den Behälter *B(0)* um. In jedem Durchlauf untersuchen wir alle Wörter auf ihre Länge. Das sind *Max_Laenge% * N%* Längentests und Listenoperationen, wobei *N%* die Anzahl zu sortierender Wörter ist. Da nur wenige Wörter die maximale Länge haben, sind die anfänglichen Operationen überflüssig.

Vorsortierung bezüglich der Wortlänge

Ziel unserer ersten Optimierung ist, die Verteilungen in den Behälter *B(0)* zu eliminieren. Wir erreichen dies, indem wir die Wörter zuerst bezüglich der Wortlänge sortieren. Anschließend verteilen wir wie gewohnt bezüglich der Zeichen. Ein Wort beziehen wir erst in die Verteilung ein, wenn *Char_Nr%* kleiner oder gleich der Wortlänge ist. Für die Längensortierung eignet sich der Behältersortieralgorithmus mit der Wortlänge als Schlüssel. Wir definieren zusätzliche Behälter *L()*. Der Behälter *L(i)* speichert eine Schlange mit den Wörtern der Länge *i*. Im ersten Durchlauf verteilen wir alle Wörter der Länge *Max_Laenge%*. Im zweiten Durchlauf ist *Char_Nr%* gleich *Max_-Laenge%-1*. Dabei ordnen wir alle Wörter der Längen *Max_Laenge%-1* und *Max_-Laenge%*. Zum Schluß sortieren wir alle Wörter. Allgemein gilt: Im i-ten Durchlauf ist *Char_Nr%* gleich *Max_Laenge%-i+1*. Wir verteilen alle Wörter, die mindestens (!) *Char_Nr%* Zeichen lang sind. Dies realisieren wir, indem wir die *Behälter L()* nicht wie gewohnt vor der Zeichensortierung aufsammeln. Stattdessen beziehen wir in jedem Durchlauf einen weiteren Behälter von *L()* in die Zeichensortierung ein. *L()* lösen wir Schritt für Schritt auf. Als Beispiel betrachten wir die Schlange

CAB, BCD, DA, BADC, C, ACBD, AB, BD

Als erstes verteilen wir die Wörter bezüglich ihrer Länge in die vier Behälter *L()*:

L(1): C
L(2): DA, AB, BD
L(3): CAB, BCD
L(4): BADC, ACBD

Nun beginnt die Sortierung nach Zeichen. Im ersten Durchlauf betrachen wir das vierte Zeichen. Wir verteilen nur die Wörter aus Behälter *L(4)* in die Behälter *B(A)*, *B(B)*, *B(C)* und *B(D)*.

B(A): leer
B(B): leer
B(C): BADC
B(D): ACBD

Nach dem Auflösen der Behälter erhalten wir die Schlange BADC, ACBD. Im zweiten Durchlauf verteilen wir alle Wörter der Längen drei und vier. Deshalb hängen wir die bezüglich des vierten Zeichens sortierte Schlange an die Schlange in *L(3)*:

CAB, BCD, BADC, ACBD

Diese vier Wörter verteilen wir bezüglich des dritten Zeichens:

B(A): leer
B(B): CAB, ACBD
B(C): leer
B(D): BCD, BADC

Die resultierende Schlange CAB, ACBD, BCD, BADC hängen wir an die Schlange aus *L(2)* und erhalten

DA, AB, BD, CAB, ACBD, BCD, BADC

Diese verteilen wir im dritten Durchlauf bezüglich des zweiten Zeichens:

B(A): DA, CAB, BADC
B(B): AB
B(C): ACBD, BCD
B(D): BD

Nach Auflesen der Behälter und Anhängen an die Schlange aus *L(1)* erhalten wir:

C, DA, CAB, BADC, AB, ACBD, BCD, BD

Diese Schlange ordnen wir abschließend bezüglich des ersten Zeichens:

B(A): AB, ACBD
B(B): BADC, BCD, BD
B(C): C, CAB
B(D): DA

Die sortierte Schlange lautet somit:

AB, ACBD, BADC, BCD, BD, C, CAB, DA

Wir haben 2+4+7+8=21 mal ein Wort in einen Behälter verteilt. Ohne Längenvorsortierung benötigen wir 4*8=32 Verteilungen - die Zahlen sprechen für sich. Diese Version von Radixsort entspricht weitgehend dem allgemeinen Radixsort aus dem dritten Kursteil. Wir betrachten neben den Zeichen auch die Länge der Wörter als Schlüssel. Die Länge hat dabei eine niedrigere Priorität als die Zeichen. Falls wir die Behälter *L* auflösen und erst dann nach der Länge sortieren, verfahren wir genau nach dem Algorithmus aus Bild

16. Allerdings müssen wir dann öfter die Wortlänge untersuchen.

Wir implementieren die Längenbehälter *L()* analog zu *B()* in zwei zusätzlichen Feldern namens *L_Anfang%()* und *L_Ende%()*.

```
DIM L_Anfang%(Max_Laenge%)
DIM L_Ende%(Max_Laenge%)
```

Der Behälter *L(i)* speichert eine Schlange mit allen Wörtern der Länge *i*.

Listing 7 zeigt die modifizierten Prozeduren *Radixsort()* und *Binsort()*. Die zusätzliche Prozedur *Laengen_Sort()* sortiert die Schlange *S* in die Behälter *L*. Die Längenbehälter sammeln wir in *Laengen_Sort()* nicht auf. *Radixsort()* hängt sie sukzessive bei der Sortierung aneinander. Bild 20 zeigt eine Hardcopy bei der Sortierung mit den Testdaten aus Listing 8. Die Druckzeilen zeigen die Schlangen, nachdem sie sortiert und mit der Liste für den nächsten Durchlauf verbunden worden sind. Dies entspricht der Ausgabe unmittelbar vor der Anweisung *NEXT Char_-Nr%* in *Radixsort()*. Der Vergleich mit Bild 19 zeigt den hohen Optimierungsgrad.

```
Zeichenketten sortieren mit Längen-Vorsortierung

Vor der Sortierung:
Bruno  Anna  Klaus  Fritz  Kunigunde  Ira  Franz  Susanne  Bernd  Emil

Sortierläufe:
15  <leere Liste>
14  <leere Liste>
13  <leere Liste>
12  <leere Liste>
11  <leere Liste>
10  Kunigunde
9  Kunigunde
8  Susanne  Kunigunde
7  Susanne  Kunigunde
6  Bruno  Klaus  Fritz  Franz  Bernd  Susanne  Kunigunde
5  Anna  Emil  Bernd  Kunigunde  Susanne  Bruno  Klaus  Fritz  Franz
4  Ira  Anna  Susanne  Kunigunde  Emil  Bernd  Bruno  Franz  Fritz  Klaus
3  Ira  Franz  Klaus  Emil  Fritz  Anna  Kunigunde  Bernd  Susanne  Bruno
2  Bernd  Klaus  Emil  Anna  Ira  Franz  Fritz  Bruno  Kunigunde  Susanne
1  Anna  Bernd  Bruno  Emil  Franz  Fritz  Ira  Klaus  Kunigunde  Susanne

Nach der Sortierung:
Anna  Bernd  Bruno  Emil  Franz  Fritz  Ira  Klaus  Kunigunde  Susanne
```

Bild 20

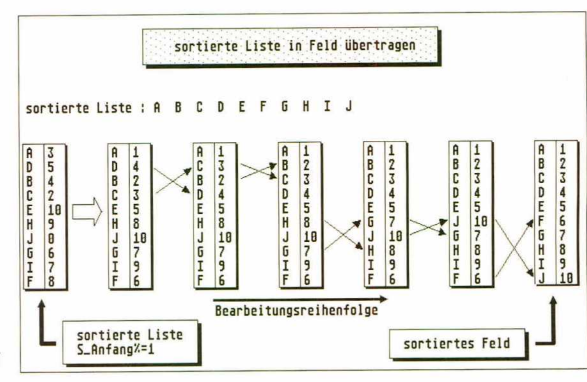


Bild 21

Auflesestrategie optimieren

In jedem Durchlauf sammeln wir alle Behälter. Viele sind oft leer, beispielsweise die für die Sonderzeichen oder Zahlen. Als nächste Optimierung minimieren wir die Anzahl aufzulesender Behälter. Ein Vorschlag (siehe <Mehlhorn>) lautet, vor der Sortierung die Zeichen zu bestimmen, die in jedem Durchlauf vorkommen. Man sammelt dann nur Behälter ein, von denen man im voraus weiß, daß sie nicht leer sind. Dieser Ansatz hat theoretische Effizienzvorteile, läßt sich aber nur mit großem Aufwand implementieren. Wir implementieren eine 'Schmalspurlösung' dieses Gedankens: In jedem Durchlauf merken wir uns den kleinsten und den größten aufgetretenen ASCII-Wert. Dazu verwenden wir die beiden Variablen *Min_Ch%* und *Max_Ch%*. Wir sammeln nur die Behälter, deren Indizes in diesem Intervall liegen. Von allen anderen Behältern wissen wir, daß sie leer sind. Es gilt

```
Min_Ascii% <= Min_Ch%
        <= Max_Ch% <= Max_Ascii%
```

Die Schleifen lauten nicht mehr

```
FOR I%=Min_Ascii% TO Max_Ascii%
```

sondern

```
FOR I%=Min_Ch% TO Max_Ch%
```

Für die Sortierung deutscher Wörter ist *Min_Ascii%* mindestens gleich 48 (bzw. 65) und *Max_Ascii%* gleich 158. Dieses Intervall umfaßt die Zeichen von '0' (bzw. 'A') bis 'ß'. Es ist realistisch, anzunehmen, daß in einem Durchlauf beispielsweise nur Kleinbuchstaben ohne Umlaute vorkommen. In diesem Fall ist *Min_Ch%* nach dem Durchlauf mindestens gleich 97 ('a'). *Max_Ch%* ist unter dieser Annahme höchstens gleich 122 ('z'). Ohne diese Optimierung löschen und sammeln sie 111 Behälter, mit Optimierung nur 26 - kleiner Trick, große Wirkung.

Für die Implementierung ändern wir die Struktur unserer Algorithmen: In *Radixsort()* löschen wir zu Beginn alle Behälter *B*. Das Behälterlöschen am Anfang von *Binsort()* entfällt, da wir die Werte für *Min_Ch%* und *Max_Ch%* noch nicht kennen. Stattdessen löschen wir die benutzten Behälter unmittelbar nach dem Auflesen. Als Aufbau erhalten wir somit:

```
Längensortierung
alle Behälter löschen
FOR alle Schlüssel DO
   Verteilen in Behälter
   benutzte Behälter aufsammeln und
                     löschen
END
```

anstelle von

```
Längensortierung
FOR alle Schlüssel DO
   alle Behälter löschen
   Verteilen in Behälter
   alle Behälter aufsammeln
END
```

Listing 8 enthält das komplette Programm zur String-Sortierung. Die Implementierung ist anschaulich und allgemein gehalten. Sie bietet eine Vielzahl von Ansätzen für effizientere Programmierung. Beispielsweise sind einige Sicherheitsabfragen bei den Listenoperationen für dieses Beispielprogramm überflüssig. In Listing 9 finden Sie das komplette Modula-2-Programm für den Public Domain-Compiler der ETH Zürich (LRU München). Es sortiert Zeichenketten mit den vorgestellten Optimierungen.

Schritt 3: sortierte Schlange in Feld übertragen

Die Sortierung der Schlange ergibt sich ausschließlich aus der Verzeigerung. Im dritten Schritt speichern wir die Daten derart um, daß sie sortiert im Feld stehen. Der zugehörige Algorithmus besteht aus zwei Teilen: Zuerst berechnen wir die neuen Feldpositionen der Wörter. Anschließend speichern wir die Wörter um. Die Position im sortierten Feld entspricht der Position in der Schlange. Beispielsweise gehört das fünfte Listenelement in

das Feldelement mit dem Index fünf. Wir berechnen diese Indizes, indem wir die Schlange durchlaufen und für jedes Element in *Wl_Next%()* die Position in der Schlange speichern. Der neue Wert in *Wl_Next%()* besagt also, wo das Wort zu speichern ist. Die Elemente sind anschließend nicht mehr als Schlange verzeigert. Die beiden linken Kästen aus Bild 21 demonstrieren diese Operation. Im zweiten Teil speichern wir die Wörter an die Feldposition, die *Wl_Next%()* angibt. Zu Beginn sucht der Algorithmus das erste Wort, das nicht an der richtigen Position steht. Im Beispiel aus Bild 21 ist das 'D'. 'D' steht an Position zwei und gehört an vier. Wir tauschen die Feldelemente zwei und vier. Jetzt befindet sich 'C' an zweiter Stelle, obwohl es an Position drei gehört. Wir tauschen also das zweite und das dritte Element. Nun steht 'B' korrekt an zweiter Stelle. Der Algorithmus sucht anschließend das nächste fehlerhaft gespeicherte Wort. Dies ist 'H' an Position sechs. Er vertauscht solange, bis an dieser Position das korrekte Wort steht. Bitte beachten Sie, daß der Algorithmus keine Worte umspeichert, die bereits korrekt gespeichert sind. Bei jeder Vertauschung schreiben wir mindestens ein Wort an seine richtige Position. Der Algorithmus terminiert nach höchstens *N%-1* Vertauschungen, da er im letzten Schritt zwei Wörter korrekt speichert. Die Prozedur *Uebertrage_Liste_In_Feld()* aus Listing 8 enthält die Implementierung.

Bewertung

Wir sortieren Strings mit dem Radixsort-Algorithmus, indem wir die Zeichen einzeln verteilen. Wir greifen auf jedes Zeichen genau einmal zu. Bei den meisten anderen Algorithmen basiert die Sortierung auf wiederholtem Vergleichen und Umspeichern von Strings. Beginnend am Anfang vergleicht man zwei Strings zeichenweise bis zur ersten Ungleichheit. Diese entscheidet, welcher der beiden in der alphabetischen Ordnung der kleinere ist. Der schnellste Vertreter dieser Sortieralgorithmen ist in Omikron.BASIC als Befehl SORT integriert: Quicksort.

Laufzeit

Wann bringt die Radixsortierung Laufzeitvorteile? Bei wenigen kurzen Strings hat Quicksort ganz klar die Nase vorn. Radixsort bietet sich an, wenn viele gleichartige Strings zu sortieren sind. Je mehr Zeichenketten sich in den letzten Zeichen unterscheiden, umso mehr eignet sich Radixsort. Konkret zu Omikron.-BASIC: Der Radixsort-Algorithmus hat gegen den effizient in Assembler programmierten SORT-Befehl keine Chance.

Umlaute

Bei Quicksort ist die korrekte Behandlung von Umlauten ('ä' als 'ae', ...) einfach zu berücksichtigen. Radixsort stellt uns in diesem Punkt vor enorme Probleme, denn die Umlaute haben völlig unbrauchbare ASCII-Werte. Als Ausweg bietet sich die Generierung eines eigenen Codes an. Bei diesem steht 'ä' unmittelbar hinter 'a'. Die Umlaute sortieren wir dann als einzelne Zeichen, das heißt 'Mueller' kommt vor 'Mühlings', anstatt umgekehrt. Fazit: Der Aufwand für die korrekte Behandlung der Umlaute ist sehr groß.

Stabilität

Ein Sortieralgorithmus heißt stabil, wenn er die Reihenfolge von Objekten mit gleichem Schlüssel nicht ändert. Angenommen, Sie sortieren Datenstrukturen mit Nachname und Vorname bezüglich des Nachnamens. Eine Sortierung ist instabil, wenn sie 'Müller, Lisa' vor 'Müller, Arndt' sortiert, obwohl vorher die richtige Reihenfolge gegeben war. Stabilität ist wichtig, da Objekte mit gleichem Schlüssel eventuell nach anderen Kriterien geordnet sein können. Radixsort ist ein stabiler Sortieralgorithmus. Genaugenommen basiert seine Korrektheit auf Stabilität. Dies haben wir am Beispiel der Spielkarten im dritten Kursteil demonstriert. Der in Omikron.BASIC implementierte Quicksort-Algorithmus ist instabil. Objekte mit gleichem Schlüssel können vertauscht werden.

Fazit

Wir sind nun am Ende dieser Artikelreihe angelangt. Wir haben Ihnen demonstriert, wie man rekursive Datenstrukturen in Omikron.BASIC und Modula-2 realisiert. Als Beispielalgorithmen haben wir Listenoperationen und Radixsort beschrieben. Sie sollten nun in der Lage sein, anhand von Standardliteratur neue Gebiete der Datenstrukturen zu erobern. Das Buch 'Algorithmen und Datenstrukturen mit Modula-2' von N. Wirth bietet beispielsweise einen guten Einstieg in Baumstrukturen. Die Implementierung in Omikron.-BASIC dürfte für Sie kein Problem mehr sein.

Sven Krüppel

Literatur:
Aho, Hopcroft, Ullman: Data Structures and Algorithms, Addison Wesley
Donald E. Knuth: The Art of Computer Programming, Vol. 3 Sorting and Searching, Addison Wesley
Kurt Mehlhorn: Datenstrukturen und effiziente Algorithmen, Band 1 Sortieren und Suchen, B.G. Teubner Stuttgart
Niklaus Wirth: Algorithmen und Datenstrukturen mit Modula-2, B.G. Teubner Stuttgart









```
1:   '**********************************************
2:   '*               Listing 6                    *
3:   '*                                            *
4:   '*     RADIXSORT zur String-Sortierung        *
5:   '*       Sortierung rein nach ASCII           *
6:   '* Version 1                                  *
7:   '* Sven Krüppel, (c) MAXON Computer 1991      *
8:   '**********************************************
9:   '*                                            *
10:  '* Es wird mit Hilfe der linearen Wortliste   *
11:  '* 'Wl' sortiert. 'Wl' speichern wir in den   *
12:  '* Arrays                                     *
13:  '*   Wl_Wort$() für die Worte (Strings) und   *
14:  '*   Wl_Next%() für die Cursor (Zeiger)       *
15:  '* 'WL' wird als Schlange interpretiert.      *
16:  '*                                            *
17:  '**********************************************
18:
19:  '**********************************************
20:  'Konstanten-Definition
21:
22:  Max_N%=100' Anzahl max. zu sortierender Worte
23:  Max_Laenge%=15'maximale Länge eines Strings
24:
25:  ' Zulässige Zeichen mittels ASCII-Codes fest-
26:  ' legen. Max_Ascii% gibt an, wieviele Behälter
27:  ' sioniert werden.
28:  Min_Ascii%= ASC("0")' kleinstes zu
29:                      ' sortierendes Zeichen
30:  Max_Ascii%= ASC("ß")' größtes zu sortierendes
31:                      ' Zeichen
32:
33:  '**********************************************
34:  'Globale Variablen: Dimensionierung und
35:  '                   Deklaration
36:
37:  'Datenstruktur Wl für Cursorrealisierung
38:  'dimensionieren
39:  DIM Wl_Wort$(Max_N%)' Feld für Worte
40:  DIM Wl_Next%(Max_N%)' Feld mit Cursor zur Ver-
41:                      ' zeigerung der linearen
42:                      ' Liste Wl.
43:  'Behälter dimensionieren. Die Behälter werden
44:  'mit den ASCII-Werten der zu sortierenden
45:  'Zeichen indiziert. Es werden mehr Behälter
46:  'angefordert, als benötigt werden.
47:  DIM B_Anfang%(Max_Ascii%)
48:  DIM B_Ende%(Max_Ascii%)
49:
50:  'N% : Anzahl zu sortierender Worte
51:  'S_Anfang% : Cursor auf den Anfang der
52:             ' Wortliste
53:  'S_Ende%   : Cursor auf das Ende der
54:             ' Wortliste
55:
56:  '**********************************************
57:  ' Sortier-Prozeduren
58:
59:  DEF PROC Radixsort(R S_Anfang%,R S_Ende%)
60:    'Radix-Sortierung der Schlange S
61:
62:    LOCAL Char_Nr%' Nummer des Zeichens, das als
63:                  ' Schlüssel benutzt wird.
64:    FOR Char_Nr%=Max_Laenge% TO 1 STEP -1
65:      Binsort(S_Anfang%,S_Ende%,Char_Nr%)
66:    NEXT Char_Nr%
67:  RETURN 'Radixsort
68:
69:
70:  DEF PROC Binsort(R S_Anfang%,R S_Ende%,
                    Char_Nr%)
71:    'Die Prozedur sortiert die Strings der
72:    'Schlange S in die Behälter.
73:    'Char_Nr% : Nummer des Zeichens im String,
74:    '           das als Schlüssel benutzt wird
75:
76:    LOCAL Wort$' Wort, das einsortiert wird
77:    LOCAL Ch%' ASCII-Wert des Zeichens, das
78:             ' einsortiert wird.
79:    LOCAL I%'  Laufvariable
80:
81:    'Behälter löschen
82:    B_Anfang%(0)=0
83:    B_Ende%(0)=0
84:    FOR I%=Min_Ascii% TO Max_Ascii%
85:      B_Anfang%(I%)=0
86:      B_Ende%(I%)=0
87:    NEXT I%
88:
89:    'S in Behälter sortieren
90:    WHILE S_Anfang%<>0
91:      P%=Wl_Next%(S_Anfang%)' Cursor auf das
92:                    ' zweite Listenelement retten
93:      Wort$=Wl_Wort$(S_Anfang%)
94:      IF LEN(Wort$)>=Char_Nr% THEN
95:        Ch%= ASC( MID$(Wort$,Char_Nr%,1))
96:      ELSE '     zu kurze Worte kommen in den
97:        Ch%=0' Behälter Nr. 0
98:      ENDIF
99:      Ll_Anhaengen(B_Anfang%(Ch%),B_Ende%(Ch%),
                     S_Anfang%)
100:     S_Anfang%=P%
101:   WEND
102:
103:   'Behälter auflösen und sortierte Liste
104:   'generieren
105:   S_Anfang%=B_Anfang%(0)
106:   S_Ende%=B_Ende%(0)
107:   FOR I%=Min_Ascii% TO Max_Ascii%
108:     IF B_Anfang%(I%)<>0 THEN ' Schlange in
109:       'Behälter B(I%) nicht leer => verbinden
110:       Ll_Verbinden(S_Anfang%,S_Ende%,
                        B_Anfang%(I%),B_Ende%(I%))
111:     ENDIF
112:   NEXT I%
113:
114: RETURN 'Binsort
```

```
1:   '**********************************************
2:   '*               Listing 7                    *
3:   '*                                            *
4:   '*     RADIXSORT zur String-Sortierung        *
5:   '*       Sortierung rein nach ASCII           *
6:   '* Version 2  -  nur Sortierprozeduren        *
7:   '* Sven Krüppel, (c) MAXON Computer 1991      *
8:   '**********************************************
9:
10:  DEF PROC Radixsort(R S_Anfang%,R S_Ende%)
11:    'Radix-Sortierung der Schlange S
12:
13:    LOCAL Char_Nr%' Nummer des Zeichens, das als
14:                  ' Schlüssel benutzt wird.
15:    Laengen_Sort(S_Anfang%,S_Ende%)
16:    FOR Char_Nr%=Max_Laenge% TO 1 STEP -1
17:      IF L_Anfang%(Char_Nr%)<>L_Ende%(Char_Nr%)
         THEN
18:        'nur sortieren, wenn mehr als ein Wort
19:        'in der Liste steht
20:        Binsort(L_Anfang%(Char_Nr%),
                   L_Ende%(Char_Nr%),Char_Nr%)
21:      ENDIF
22:      Ll_Verbinden(L_Anfang%(Char_Nr%-1),
                      L_Ende%(Char_Nr%-1),
                      L_Anfang%(Char_Nr%),
                      L_Ende%(Char_Nr%))
23:    NEXT Char_Nr%
24:    S_Anfang%=L_Anfang%(0)
25:    S_Ende%=L_Ende%(0)
26:  RETURN 'Radixsort
27:
28:  DEF PROC Laengen_Sort(R S_Anfang%,R S_Ende%)
29:    'Die Prozedur sortiert die Schlange S in die
30:    'Behälter L(). Der Schlüssel ist die Wort-
31:    'länge. Die Behälter speichern nach dem Auf-
32:    'ruf die Schlangen der entsprechenden Wort-
33:    'länge. Sie werden nicht aufgelöst.
34:    'S_Anfang% und S_Ende% sind nach dem Aufruf
35:    'undefiniert, da die Schlange S nicht mehr
36:    'existiert.
37:
38:    LOCAL L%' Wortlänge
39:    LOCAL I%' Laufvariable
40:    LOCAL P%' Hilfscursor
41:
42:    'Behälter L löschen
43:    FOR I%=0 TO Max_Laenge%
44:      L_Anfang%(I%)=0
45:      L_Ende%(I%)=0
46:    NEXT I%
47:
48:    'S in Behälter sortieren
```
→

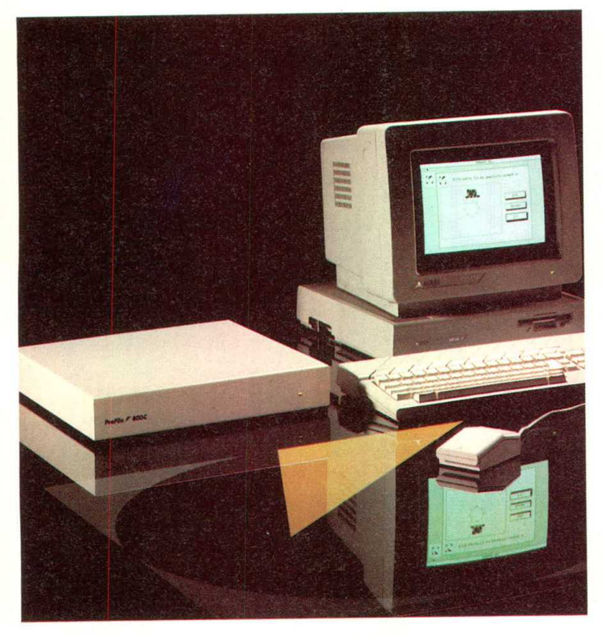
```
49:      WHILE S_Anfang%<>0
50:        P%=Wl_Next%(S_Anfang%)' Cursor auf das
51:                  ' zweite Listenelement retten
52:        L%= LEN(Wl_Wort$(S_Anfang%))
53:        Ll_Anhaengen(L_Anfang%(L%),L_Ende%(L%),
                        S_Anfang%)
54:        S_Anfang%=P%
55:      WEND
56:    RETURN 'Laengen_Sort
57:
58:
59:    DEF PROC Binsort(R S_Anfang%,R S_Ende%,
                        Char_Nr%)
60:      'Die Prozedur sortiert die Strings der
61:      'Schlange S in die Behälter.
62:      'Char_Nr% : Nummer des Zeichens im String,
63:      '           das als Schlüssel benutzt wird
64:
65:      LOCAL Wort$' Wort, das einsortiert wird
66:      LOCAL Ch%' ASCII-Code des Zeichens, das
67:               ' einsortiert wird.
68:      LOCAL I%'  Laufvariable
69:
70:      'Behälter löschen
71:      FOR I%=Min_Ascii% TO Max_Ascii%
72:        B_Anfang%(I%)=0
73:        B_Ende%(I%)=0
74:      NEXT I%
75:
76:      'S in Behälter sortieren
77:      WHILE S_Anfang%<>0
78:        P%=Wl_Next%(S_Anfang%)' Cursor auf das
79:                  ' zweite Listenelement retten
80:        Wort$=Wl_Wort$(S_Anfang%)
81:        Ch%= ASC( MID$(Wort$,Char_Nr%,1))
82:        Ll_Anhaengen(B_Anfang%(Ch%),B_Ende%(Ch%),
                        S_Anfang%)
83:        S_Anfang%=P%
84:      WEND
85:
86:      'Behälter auflösen und sortierte Liste
87:      'generieren
88:      S_Anfang%=0
89:      S_Ende%=0
90:      FOR I%=Min_Ascii% TO Max_Ascii%
91:        IF B_Anfang%(I%)<>0 THEN 'Schlange in
92:          'Behälter B(I%) nicht leer => verbinden
93:          Ll_Verbinden(S_Anfang%,S_Ende%,
                        B_Anfang%(I%),B_Ende%(I%))
94:        ENDIF
95:      NEXT I%
96:
97:    RETURN 'Binsort
```

```
1:   '**********************************************
2:   '*                Listing 8                    *
3:   '*                                             *
4:   '*      RADIXSORT zur String-Sortierung        *
5:   '*         Sortierung rein nach ASCII          *
6:   '*                                             *
7:   '*     Endversion inklusive Testumgebung       *
8:   '*                                             *
9:   '* Sven Krüppel, (c) MAXON Computer 1991       *
10:  '**********************************************
11:  '*                                             *
12:  '* Es wird mit Hilfe der linearen Wortliste    *
13:  '* 'Wl' sortiert. 'Wl' speichern wir in den    *
14:  '* Arrays                                      *
15:  '*   Wl_Wort$() für die Worte (Strings) und    *
16:  '*   Wl_Next%() für die Cursor (Zeiger)        *
17:  '* 'WL' wird als Schlange interpretiert.       *
18:  '*                                             *
19:  '**********************************************
20:
21:  '**********************************************
22:  'Konstanten-Definition
23:
24:  Max_N%=100' Anzahl max. zu sortierender Worte
25:  Max_Laenge%=15'maximale Länge eines Strings
26:
27:  ' Zulässige Zeichen mittels ASCII-Codes fest-
28:  ' legen. Max_Ascii% gibt an, wieviele Behälter
29:  ' sioniert werden.
30:  Min_Ascii%= ASC("0")' kleinstes zu
31:                        ' sortierendes Zeichen
32:  Max_Ascii%= ASC("ß")' größtes zu sortierendes
33:                        ' Zeichen
34:
35:  '**********************************************
36:  'Globale Variablen: Dimensionierung und
37:  '                   Deklaration
38:
39:  'Datenstruktur Wl für Cursorrealisierung
40:  'dimensionieren
41:  DIM Wl_Wort$(Max_N%)' Feld für Worte
42:  DIM Wl_Next%(Max_N%)' Feld mit Cursor zur Ver-
43:                        ' zeigerung der linearen
44:                        ' Liste Wl.
45:  'Behälter dimensionieren. Die Behälter werden
46:  'mit den ASCII-Codes der zu sortierenden
47:  'Zeichen indiziert. Es werden mehr Behälter
48:  'angefordert, als benötigt werden.
49:  DIM B_Anfang%(Max_Ascii%)
50:  DIM B_Ende%(Max_Ascii%)
51:
52:  'Behälter für Sortierung nach Wortlänge
53:  'dimensionieren. Behälter i zeigt auf eine
54:  'lineare Liste, die Worte der Länge i ent-
55:  'hält.
56:  DIM L_Anfang%(Max_Laenge%)
57:  DIM L_Ende%(Max_Laenge%)
58:
59:  'N% : Anzahl zu sortierender Worte
60:  'S_Anfang% : Cursor auf den Anfang der
61:              ' Wortliste
62:  'S_Ende%   : Cursor auf das Ende der
63:              ' Wortliste
64:
65:  '**********************************************
66:  'Hauptprogramm
67:
68:  Wl_Wort$(1)="Bruno"
69:  Wl_Wort$(2)="Anna"
70:  Wl_Wort$(3)="Klaus"
71:  Wl_Wort$(4)="Fritz"
72:  Wl_Wort$(5)="Kunigunde"
73:  Wl_Wort$(6)="Ira"
74:  Wl_Wort$(7)="Franz"
75:  Wl_Wort$(8)="Susanne"
76:  Wl_Wort$(9)="Bernd"
77:  Wl_Wort$(10)="Emil"
78:
79:  N%=10' Anzahl zu sortierender Worte
80:  Generiere_Liste_Aus_Feld(N%,S_Anfang%,S_Ende%)
81:  Radixsort(S_Anfang%,S_Ende%)
82:  Ll_Ausgeben(S_Anfang%)
83:  Uebertrage_Liste_In_Feld(S_Anfang%,S_Ende%,N%)
84:  Wl_Wort_Ausgeben(N%)
85:  END
86:
87:  '**********************************************
88:  ' Sortier-Prozeduren
89:
90:  DEF PROC Radixsort(R S_Anfang%,R S_Ende%)
91:    'Radix-Sortierung der Schlange S
92:
93:    LOCAL Char_Nr%' Nummer des Zeichens, das als
94:                  ' Schlüssel benutzt wird.
95:
96:    Laengen_Sort(S_Anfang%,S_Ende%)
97:    B_Init(Min_Ascii%,Max_Ascii%)
98:    'Sortieren
99:    FOR Char_Nr%=Max_Laenge% TO 1 STEP -1
100:     IF L_Anfang%(Char_Nr%)<>L_Ende%(Char_Nr%)
       THEN
101:        'nur sortieren, wenn mehr als ein Wort
102:        'in der Liste steht
103:        Binsort(L_Anfang%(Char_Nr%),
                  L_Ende%(Char_Nr%),Char_Nr%)
104:     ENDIF
105:     Ll_Verbinden(L_Anfang%(Char_Nr%-1),
                    L_Ende%(Char_Nr%-1),
                    L_Anfang%(Char_Nr%),
                    L_Ende%(Char_Nr%))
106:   NEXT Char_Nr%
107:   S_Anfang%=L_Anfang%(0)
108:   S_Ende%=L_Ende%(0)
109: RETURN 'Radixsort
110:
111:
112: DEF PROC Laengen_Sort(R S_Anfang%,R S_Ende%)
```

→

protar ... mehr als nur Festplatten











Eine komplette Produktfamilie für alle Atari-Freunde. Ob Profi oder Amateur, ob Konstrukteur oder Künstler, ob Autor oder Spielefan – jeder findet hier die optimale Lösung. Für jede Anwendung, für jeden Geldbeutel.

1. Beispiel: die ProFile SCSI-Festplatten. 20 bis 440 MB. Booten von beliebigen Partitionen. Bis zu 12 Partitionen je Platte, frei wählbar. Schreib- und Passwortschutz.
2. Beispiel: die ProFile SCSI-Tape Streamer T60/T150. 60 oder 150 MB. Komfortable Backup-Software. Eigener Desktop, Batch-Sprache.
3. Beispiel: das ProFile R44 Wechselplattenlaufwerk. Es kombiniert die Vorteile einer 44MB protar-Festplatte mit der einfachen Handhabung von Disketten.
4. Beispiel: der ProScreenTT. 19" Großbildschirm für den Atari TT. WS Phosphor (paperwhite) Bildröhre. 72 Hz Bildwiederholrate. Auflösung 1280 x 960. Dreh- und Schwenkfuß integriert.

protar – wir machen Qualität preiswert

Alle genannten Geräte bieten Ihnen die Zuverlässigkeit und die Flexibilität, die Sie als Atari-User heute brauchen.

Die Festplatten sind schnell und leise. Die Monitore ermöglichen Ihnen ein ermüdungsfreies Arbeiten.

Die Kompatibilität zu allen Atari-Komponenten und allen gängigen Software-Paketen ist selbstverständlich.

Informationen zu protar-Produkten erhalten Sie bei Ihrem qualifizierten Fachhändler.

protar Elektronik GmbH Alt-Moabit 91 D 1000 Berlin 21 Telefon 030 391 20 02 Fax 030 391 73 32

Vertretung in Österreich: Dipl. Ing. R. Temmel Ges. mbH & Co KG St. Julienstraße 4a 5020 Salzburg Telefon 0662 71 81 64 Fax 06244 71 88 3

Vertretung in der Schweiz: DTZ Data Trade AG Landstraße 1 CH 5415 Rieden/Baden Telefon 056 82 18 80 Fax 056 82 18 84

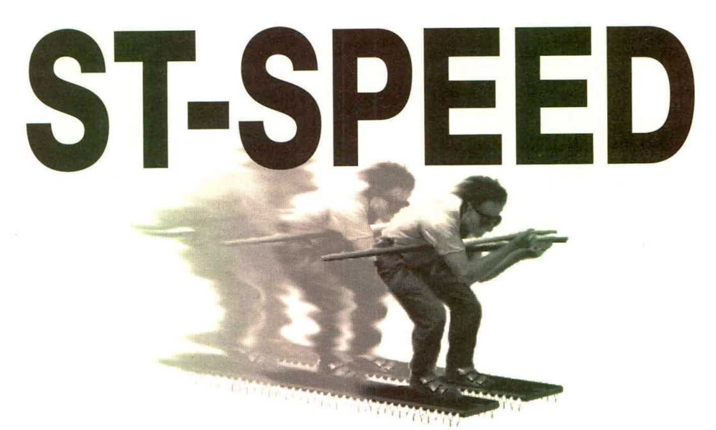
```
113:       'Die Prozedur sortiert die Schlange S in die
114:       'Behälter L(). Der Schlüssel ist die Wort-
115:       'länge. Die Behälter speichern nach dem Auf-
116:       'ruf die Schlangen der entsprechenden Wort-
117:       'länge. Sie werden nicht aufgelöst.
118:       'S_Anfang% und S_Ende% sind nach dem Aufruf
119:       'undefiniert, da die Schlange S nicht mehr
120:       'existiert.
121:       LOCAL L%' Wortlänge
122:       LOCAL I%' Laufvariable
123:       LOCAL P%' Hilfscursor
124:
125:       'Behälter L löschen
126:       FOR I%=0 TO Max_Laenge%
127:         L_Anfang%(I%)=0
128:         L_Ende%(I%)=0
129:       NEXT I%
130:
131:       'S in Behälter sortieren
132:       WHILE S_Anfang%<>0
133:         P%=Wl_Next%(S_Anfang%)' Cursor auf das
134:                      ' zweite Listenelement retten
135:         L%= LEN(Wl_Wort$(S_Anfang%))
136:         Ll_Anhaengen(L_Anfang%(L%),L_Ende%(L%),
                          S_Anfang%)
137:         S_Anfang%=P%
138:       WEND
139:     RETURN 'Laengen_Sort
140:
141:
142:     DEF PROC B_Init(Min_Ch%,Max_Ch%)
143:       'Behälter B löschen
144:       FOR I%=Min_Ch% TO Max_Ch%
145:         B_Anfang%(I%)=0
146:         B_Ende%(I%)=0
147:       NEXT I%
148:     RETURN 'B_Loeschen
149:
150:
151:     DEF PROC Binsort(R S_Anfang%,R S_Ende%,
                        Char_Nr%)
152:       'Die Prozedur sortiert die Strings der
153:       'Schlange S in die Behälter.
154:       'Char_Nr% : Nummer des Zeichens im String,
155:       '           das als Schlüssel benutzt wird
156:
157:       LOCAL Wort$' Wort, das einsortiert wird
158:       LOCAL Ch%' ASCII-Wert des Zeichens, das
159:              ' einsortiert wird.
160:       LOCAL I%'  Laufvariable
161:       LOCAL Min_Ch%,Max_Ch%' ASCII-Werte des
162:                            ' kleinsten und größten
163:                            ' benutzten Zeichens
164:
165:       'S in Behälter sortieren
166:       Min_Ch%=Max_Ascii%' beginne mit dem schlech-
167:       Max_Ch%=Min_Ascii%' testen Fall
168:       WHILE S_Anfang%<>0
169:         P%=Wl_Next%(S_Anfang%)' Cursor auf das
170:                      ' zweite Listenelement retten
171:         Wort$=Wl_Wort$(S_Anfang%)
172:         Ch%= ASC( MID$(Wort$,Char_Nr%,1))
173:         Min_Ch%= MIN(Min_Ch%,Ch%)
174:         Max_Ch%= MAX(Max_Ch%,Ch%)
175:         Ll_Anhaengen(B_Anfang%(Ch%),B_Ende%(Ch%),
                          S_Anfang%)
176:         S_Anfang%=P%
177:       WEND
178:
179:       'Behälter auflösen und sortierte Liste
180:       'generieren. Inklusive Behälter löschen
181:       S_Anfang%=0
182:       S_Ende%=0
183:       FOR I%=Min_Ch% TO Max_Ch%
184:         IF B_Anfang%(I%)<>0 THEN
185:           'Behälter nicht leer
186:           Ll_Verbinden(S_Anfang%,S_Ende%,
                          B_Anfang%(I%),B_Ende%(I%))
187:           B_Anfang%(I%)=0' Behälter löschen
188:           B_Ende%(I%)=0
189:         ENDIF
190:       NEXT I%
191:
192:     RETURN 'Binsort
193:
194:
195:     '*********************************************
196:     '  Prozeduren für die Listenoperationen
197:
198:     DEF PROC Ll_Anhaengen(R Anfang%,R Ende%,P%)
199:       'Hängt das Element, auf das P% zeigt, an das
200:       'Ende der Liste an, die durch Anfang% und
201:       'Ende% charakterisiert ist.
202:       '(!) Die Prozedur enthält keine Schutz-
203:       'Mechanismen gegen fehlerhafte Übergabe-
204:       'parameter
205:       IF Anfang%=0 THEN Anfang%=P%' Liste war
206:                                   ' bislang leer
207:       Wl_Next%(P%)=0' Ende der verlängerten
208:                     ' Liste markieren
209:       Wl_Next%(Ende%)=P%' neues Element anhängen
210:       Ende%=P%'          Cursor auf das Listenende
211:                        ' umbiegen
212:     RETURN 'Ll_Anhaengen
213:
214:
215:     DEF PROC Ll_Verbinden(R Anfang1%,R Ende1%,
                             Anfang2%,Ende2%)
216:       'Hängt die durch Anfang2% und Ende2%
217:       'definierte Liste an die durch Anfang1% und
218:       'Ende1% definierte Liste.
219:       IF Anfang1%=0 THEN '   1. Liste ist leer.
220:         Anfang1%=Anfang2%' funktioniert auch,
221:         Ende1%=Ende2%'     wenn beide leer
222:       ELSE ' 1. Liste ist nicht leer
223:         IF Anfang2%<>0 THEN '2. Liste nicht leer
224:           Wl_Next%(Ende1%)=Anfang2%' verbinden
225:           Ende1%=Ende2%
226:         ENDIF
227:       ENDIF
228:     RETURN 'Ll_Verbinden
229:
230:
231:     DEF PROC Ll_Ausgeben(Anfang%)
232:       'Druckt die Liste ab Index Anfang%
233:       IF Anfang%=0 THEN PRINT "<leere Liste>";
234:       WHILE Anfang%<>0
235:         PRINT Wl_Wort$(Anfang%);"  ";
236:         Anfang%=Wl_Next%(Anfang%)
237:       WEND
238:       PRINT
239:     RETURN 'Ll_Ausgeben
240:
241:
242:     '*********************************************
243:     ' Konvertierung: Feld <-> Liste
244:
245:     DEF PROC Generiere_Liste_Aus_Feld(N%,
                                R S_Anfang%, R S_Ende%)
246:       ' Die Prozedur generiert eine lineare Liste,
247:       ' indem sie die Cursor in 'Wl_Next%()'
248:       ' initialisiert.
249:       FOR I%=1 TO N%-1
250:         Wl_Next%(I%)=I%+1' jeder zeigt auf seinen
251:       NEXT I%'              Nachfolger
252:       Wl_Next%(N%)=0' Listenende markieren
253:       S_Anfang%=1
254:       S_Ende%=N%
255:     RETURN 'Generiere_Liste_Aus_Feld
256:
257:
258:     DEF PROC Uebertrage_Liste_In_Feld(R S_Anfang%,
                                   R S_Ende%,R N%)
259:       ' Die Prozedur organisiert die Wörter derart,
260:       ' daß sie sortiert im Feld stehen.
261:       ' In N% wird die Anzahl der Wörter zurück-
262:       ' gegeben.
263:       ' Die Wörter werden in die Elemente 1 bis
264:       ' N% von Wl_Wort$() geschrieben.
265:       ' Die Listenstruktur wird zerstört, da die
266:       ' Cursor in Wl_Next% überschrieben werden.
267:       ' S_Anfang% und S_Ende% werden gelöscht
268:     LOCAL N%' Anzahl
269:     LOCAL I%,J%' Hilfsvariablen
270:
271:     ' Feld-Position der Listenelemente in Wl_Next%
272:     ' eintragen.
273:     N%=0
274:     I%=S_Anfang%' Cursor
275:     REPEAT
276:       N%=N%+1' Anzahl der Wörter inkrementieren
277:       J%=Wl_Next%(I%)' Cursor auf den Nachfolger
278:                      ' retten
279:       Wl_Next%(I%)=N%' Feldposition eintragen  →
```

```
280:     I%=J%' Cursor auf das nächste Listenelement
281:   UNTIL I%=0' bis Listenende
282:
283:   'alle Wörter umspeichern
284:   I%=1' Position, bis zu der die Wörter sortiert
285:       ' im Feld stehen.
286:   'nächstes fehlerhaft gespeichertes Wort suchen
287:   WHILE Wl_Next%(I%)=I%
288:     I%=I%+1
289:   WEND
290:   REPEAT
291:     WHILE Wl_Next%(I%)<>Wl_Next%(Wl_Next%(I%))
292:       'solange Wörter vertauschen, bis an Pos.
293:       'I% das richtige Wort steht
294:       SWAP Wl_Wort$(I%),Wl_Wort$(Wl_Next%(I%))
295:       SWAP Wl_Next%(I%),Wl_Next%(Wl_Next%(I%))
296:     WEND
297:     I%=I%+1' mindestens ein Wort mehr steht nun
298:            ' an der richtigen Position
299:     'nächstes fehlerhaft gespeichertes Wort
300:     'suchen
301:     WHILE Wl_Next%(I%)=I%
302:       I%=I%+1
303:     WEND
304:   UNTIL I%>N%
305:   S_Anfang%=0:S_Ende%=0
306:   RETURN 'Uebertrage_Liste_In_Feld
307:
308:
309:   '*********************************************
310:   ' Testprozeduren
311:
312:   DEF PROC B_Ausgeben(Min_Ch%,Max_Ch%)
313:   ' Behälterfeld B() ausgeben
314:     LOCAL I%
315:
316:     FOR I%=Min_Ch% TO Max_Ch%
317:       Ll_Ausgeben(B_Anfang%(I%))
318:     NEXT I%
319:   RETURN 'B_Ausgeben
320:
321:   DEF PROC L_Ausgeben
322:   ' Liste ausgeben
323:     LOCAL I%
324:
325:     FOR I%=0 TO Max_Laenge%
326:       PRINT I%;"  ";:Ll_Ausgeben(L_Anfang%(I%))
327:     NEXT I%
328:   RETURN 'L_Ausgeben
329:
330:   DEF PROC Wl_Wort_Ausgeben(N%)
331:   ' Wörterfeld Wl_Wort$() ausgeben
332:     LOCAL I%
333:
334:     FOR I%=0 TO N%
335:       PRINT Wl_Wort$(I%)
336:     NEXT I%
```

```
  1:  (**********************************************
  2:  *                Listing 9                    *
  3:  *   Radixsort für Zeichenketten-Sortierung    *
  4:  *       Sortierung nach ASCII-Werten          *
  5:  *                                             *
  6:  *   Sven Krüppel  (c) MAXON Computer 1991     *
  7:  *   geschrieben mit dem ETH-Modula 2-System   *
  8:  **********************************************)
  9:
 10:  MODULE Listing9;
 11:
 12:  FROM InOut IMPORT Write,WriteString,WriteLn;
 13:  FROM Heap IMPORT Allocate, Deallocate;
 14:  FROM Strings IMPORT Length;
 15:
 16:  CONST MaxN=100; (* Anzahl max. zu sortierender
 17:                     Records *)
 18:        MinKey="0"; (* kleinster Schlüsselwert *)
 19:        MaxKey="ß"; (* größter Schlüsselwert  *)
 20:        MaxLaenge=15; (* max. Stringlänge *)
 21:
 22:  TYPE String = ARRAY [1..MaxLaenge] OF CHAR;
 23:       KeyTyp = [MinKey..MaxKey]; (* Wertebereich
 24:                              der Schlüsselwerte *)
 25:       ObjTyp = RECORD  (* Record für Daten *)
 26:                  Wort : String; (* Wörter *)
 27:                  (* <weitere Infokomponenten> *)
 28:                END;
 29:       ObjFTyp= ARRAY [1..MaxN] OF ObjTyp;
 30:       ObjLPtr= POINTER TO ObjLTyp;
 31:       ObjLTyp= RECORD (* Schlangenelement *)
 32:                  Obj  : ObjTyp;
 33:                  Next : ObjLPtr;
 34:                END;
 35:       STyp   = RECORD (* Schlangen-, Listentyp*)
 36:                  Anfang : ObjLPtr;
 37:                  Ende   : ObjLPtr;
 38:                END;
 39:
 40:  VAR  N : CARDINAL; (* Anzahl zu sortierender
 41:                        Records *)
 42:      TF : ObjFTyp;  (* Feld mit Testdaten *)
 43:      TL : STyp;     (* Liste mit Testdaten *)
 44:       p : ObjLPtr;  (* Hilfszeiger *)
 45:       i : CARDINAL; (* Laufvariable *)
 46:
 47:
 48:  (*********************************************
 49:  *  Element an eine lineare Liste anhängen    *
 50:  *  Dies entspricht dem Einfügen in eine      *
 51:  *  Schlange                                  *
 52:  *********************************************)
 53:
 54:  PROCEDURE LLAnhaengen(VAR L:STyp; P:ObjLPtr);
 55:  (* Element, auf das 'P' zeigt, an die lineare
 56:     Liste 'L' hängen. Diese Operation ent-
 57:     spricht dem Einfügen in eine Schlange.
 58:  *)
 59:  BEGIN
 60:    IF P <> NIL THEN
 61:      P^.Next:=NIL;    (* neues Listenende
 62:                          markieren *)
 63:      IF L.Anfang=NIL THEN (* Liste ist leer *)
 64:        L.Anfang:=P;
 65:      ELSE (* Liste war nicht leer *)
 66:        L.Ende^.Next:=P;(* Element anhängen,
 67:                           auf das P zeigt *)
 68:      END;
 69:      L.Ende:=P;   (* Zeiger auf das
 70:                      Listenende umbiegen *)
 71:    END (* IF *)
 72:  END LLAnhaengen;
 73:
 74:  (*********************************************
 75:  *  Zwei Listen verbinden                     *
 76:  *********************************************)
 77:
 78:  PROCEDURE LLVerbinden(VAR L1, L2: STyp);
 79:  (* Liste 'L2' wird an Liste 'L1' gehängt. Die
 80:     Ergebnisliste ist 'L1'.
 81:  *)
 82:  BEGIN
 83:  IF L1.Anfang=NIL THEN (* 1. Liste ist leer *)
 84:     L1:=L2;
 85:  ELSIF L2.Anfang<>NIL THEN (* 2. Liste nicht
 86:                               leer *)
 87:     L1.Ende^.Next:=L2.Anfang; (* verbinden *)
 88:     L1.Ende:=L2.Ende
 89:  END;
 90:  END LLVerbinden;
 91:
 92:  (*********************************************
 93:  *  Liste ausgeben                            *
 94:  *********************************************)
 95:
 96:  PROCEDURE LLAusgeben(L : STyp);
 97:  VAR p : ObjLPtr;
 98:  BEGIN
 99:    p:=L.Anfang;
100:    WHILE p<>NIL DO
101:      WriteString(p^.Obj.Wort);
102:      WriteLn;
103:      p:=p^.Next;
104:    END(*WHILE*);
105:    WriteLn;
106:  END LLAusgeben;
107:
108:
109:  (*********************************************
110:  *                                            *
111:  *              Radix-Sortierung              *
112:  *                                            *
113:  * Die Prozedur RadixSort enthält mehrere *
```

```
114:  * lokale Prozeduren.                          *
115:  *********************************************)
116:
117:  PROCEDURE RadixSort(VAR S : STyp);
118:
119:  VAR B : ARRAY KeyTyp OF STyp; (* Feld mit
120:                                Behältern *)
121:      L : ARRAY [0..MaxLaenge] OF STyp; (* Feld
122:                          mit Behältern für die
123:                          Längensortierung *)
124:      CharNr : CARDINAL; (* Position des Zeichens,
125:                            das als Schlüssel be-
126:                            nutzt wird *)
127:
128:  (*********************************************
129:  *           lokale Prozeduren                 *
130:  *********************************************)
131:
132:
133:  (***********  Längen-Sortierung *************)
134:
135:  PROCEDURE LaengenSort(VAR S : STyp);
136:  (* Die Prozedur sortiert die Strings der
137:     Schlange S bezüglich der Wortlänge
138:     in die Behälter L. Die Behälter speichern
139:     nach dem Aufruf die Schlangen der entspre-
140:     chenden Wortlänge. Sie werden nicht aufge-
141:     löst. S ist nach dem Aufruf undefiniert,
142:     da die Schlange S nicht mehr existiert.
143:  *)
144:  VAR p : ObjLPtr;  (* Hilfszeiger *)
145:      i : CARDINAL; (* Laufvariable *)
146:
147:  BEGIN
148:   (* Behälter L löschen *)
149:   FOR i:=0 TO MaxLaenge DO
150:     L[i].Anfang:=NIL;
151:     L[i].Ende:=NIL;
152:   END;
153:
154:   (* S in die Behälter sortieren *)
155:   (* nähere Erläuterungen bei Binsort() *)
156:   WHILE S.Anfang<>NIL DO
157:     p:=S.Anfang^.Next; (* Zeiger auf nächstes
158:                           Element retten *)
159:     LLAnhaengen(L[Length(S.Anfang^.Obj.Wort)],
160:                 S.Anfang);
161:     S.Anfang:=p;
162:   END;
163:  END LaengenSort;
164:
165:
166:  (********* Behälter B initialisieren ********)
167:  PROCEDURE BInit(MinCh, MaxCh : KeyTyp);
168:  VAR i : KeyTyp;
169:  BEGIN
170:   FOR i:=MinKey TO MaxKey DO
171:     B[i].Anfang:=NIL;
172:     B[i].Ende:=NIL;
173:   END;
174:  END BInit;
175:
176:
177:  (***********  Behältersortieren *************)
178:
179:  PROCEDURE BinSort(VAR S : STyp;
180:                    CharNr : CARDINAL);
181:  (* Die Prozedur sortiert die Strings der
182:     Schlange S bezüglich des CharNr-ten Zeichens
183:     in die Behälter B.
184:  *)
185:  VAR p : ObjLPtr; (* Hilfszeiger *)
186:      Char,         (* Hilfsvariable *)
187:      MinChar, MaxChar, (* kleinstes bzw. größtes
188:                          aufgetretenes Zeichen *)
189:      i : KeyTyp;  (* Laufvariable *)
190:
191:  BEGIN
192:
193:   (* Schlange S in Behälter sortieren *)
194:   (* Es wird jeweils der Listenanfang von S
195:      in einen Behälter einsortiert.
196:      LLAnhaengen zerstört den Next-Zeiger
197:      des Listenanfangs. Deshalb muß ein Zeiger
198:      auf das jeweils zweite Listenelement
199:      zwischengespeichert werden.
200:   *)
201:   MinChar:=MaxKey; (* beginne mit dem     *)
202:   MaxChar:=MinKey; (* schlechtesten Fall *)
203:   WHILE S.Anfang<>NIL DO
204:     p:=S.Anfang^.Next; (* Zeiger auf nächstes
205:                           Element retten *)
206:     Char:=S.Anfang^.Obj.Wort[CharNr];
207:     IF Char<MinChar THEN MinChar:=Char END;
208:     IF Char>MaxChar THEN MaxChar:=Char END;
209:     LLAnhaengen(B[Char], S.Anfang);
210:     S.Anfang:=p;
211:   END;
212:
213:   (* Behälter auflösen und sortierte
214:      Liste generieren. Dabei werden die
215:      Behälter gelöscht.
216:   *)
217:   S.Anfang:=NIL;
218:   S.Ende:=NIL;
219:   FOR i:=MinKey TO MaxKey DO
220:     IF B[i].Anfang<>NIL THEN (* Schlange in
221:       Behälter B[i] nicht leer => verbinden *)
222:       LLVerbinden(S,B[i]);
223:       B[i].Anfang:=NIL;
224:       B[i].Ende:=NIL;
225:     END;
226:   END;
227:  END BinSort;
228:
229:
230:  (*********************************************
231:  *      Hauptprozedur : RadixSort            *
232:  *********************************************)
233:
234:  BEGIN (* RadixSort *)
235:    LaengenSort(S);
236:
237:    BInit(MinKey, MaxKey);
238:    FOR CharNr:=MaxLaenge TO 1 BY -1 DO
239:      IF L[CharNr].Anfang # L[CharNr].Ende THEN
240:        (* nur sortieren, wenn die Schlange mehr
241:           als ein Wort enthält *)
242:        BinSort(L[CharNr], CharNr);
243:      END;
244:      LLVerbinden(L[CharNr-1], L[CharNr]);
245:    END(*FOR*);
246:    S.Anfang:=L[0].Anfang;
247:    S.Ende:=L[0].Ende;
248:  END RadixSort;
249:
250:  (*********************************************
251:  *            Testliste generieren           *
252:
253:  PROCEDURE ErzeugeListeAusFeld(F : ObjFTyp;
254:            VAR L : STyp);
255:  (* Die Liste L wird zurückgegeben *)
256:  VAR p : ObjLPtr; (* Hilfszeiger *)
257:      i : CARDINAL;
258:  BEGIN
259:   L.Anfang:=NIL;
260:   L.Ende:=NIL;
261:   FOR i:=1 TO N DO
262:     Allocate(p,SIZE(ObjLTyp));
263:     p^.Obj.Wort:=F[i].Wort;
264:     LLAnhaengen(L,p);
265:   END(*FOR*);
266:  END ErzeugeListeAusFeld;
267:  BEGIN (* Hauptprogramm *)
268:    (* Feld mit Testdaten initialisieren *)
269:    TF[1].Wort:="Bruno";
270:    TF[2].Wort:="Anna";
271:    TF[3].Wort:="Klaus";
272:    TF[4].Wort:="Fritz";
273:    TF[5].Wort:="Kunigunde";
274:    TF[6].Wort:="Ira";
275:    TF[7].Wort:="Franz";
276:    TF[8].Wort:="Susanne";
277:    TF[9].Wort:="Bernd";
278:    TF[10].Wort:="Emil";
279:
280:    N:=10;
281:    (* Liste mit Testdaten generieren *)
282:    ErzeugeListeAusFeld(TF,TL);
283:    LLAusgeben(TL);
284:    RadixSort(TL); (* Liste sortieren *)
285:    LLAusgeben(TL);
286:
287:  END Listing9.
```

ST-SPEED



Flexibles Utility Teil 2

Heute geht es an die RAM-Disk-Routinen und den XBRA-Lister. Erstere erlauben u.a. auch den Schreibschutz der RAM-Disk; ob dieses sinnvoll ist oder nicht, bleibt jedem selber überlassen; wichtig ist, daß Sie auch die Partitionen Ihrer Harddisk schreibschützen können.

Die RAM-Disk

Den längsten Teil benötigt die Installation der RAM-Disk (Zeilen 380 bis 550). Danach folgen die Routinen *HDV_BPB*, *HDV_RW* und *HDV_MEDIA*.

Es kann jeweils nur ein RAM-Disk-Laufwerk verwaltet werden (wer braucht schon zwei, oder sogar noch mehr RAM-Disks?). Über das Flag *INSTALLFLAG* kann erfragt werden, ob bereits eine RAM-Disk installiert wurde (Flag<>0) oder nicht. Ist eine Disk bereits vorhanden, wird ins Reinstallationsmenü verzweigt, ansonsten erfolgt die Installation.

Das Flag RESIDENT=0 zeigt an, daß die RAM-Disk nicht reset-resident sein soll.

Da unsere RAM-Disk nicht einfach nur das nächste freie Laufwerk belegen soll, sondern das Laufwerk einen beliebigen Laufwerksbuchstaben haben kann, kann man diesen ebenfalls ändern. Das ASCII-Zeichen steht in der Variablen *DRIVECHAR* und in *DRIVENR* die Laufwerksnummer. Da das Utility möglichst kurz sein sollte, wurde auf eine komfortable Eingabe der RAM-Disk-Größe verzichtet. Eine Eingaberoutine hätte nur unnötig Platz belegt, deshalb wird die gedrückte Ziffer einfach mit 100 multipliziert (die Null entspricht der 10), und die so erhaltene Zahl gibt die Speichergröße in kBytes an.

Ab Zeile 499 werden die Gesamtgröße der RAM-Disk bestimmt und der Speicherbereich reserviert. Zum Schluß werden nur noch die ersten Sektoren, der RAM-Disk gelöscht, ein freies Laufwerk gesucht und die Vektoren installiert.

Wenn bereits eine RAM-Disk vorhanden war, wird ab Zeile 439 nachgefragt, ob diese gelöscht werden soll. Wenn diese nicht mehr benötigt wird, wird die Laufwerkskennung gelöscht und die Vektoren wieder zurückgesetzt.

Die RAM-Disk-Routinen (ab Zeile 566) beginnen zunächst mit der BPB-Routine. Da diese Routine nur aufgerufen werden darf, wenn auch eine RAM-Disk installiert ist, wird ggf. in Zeile 556 noch abgebrochen. Auch zur MEDIA-Routine gibt es nicht viel zu sagen. Da man eine RAM-Disk nicht wechseln kann (wer's schon einmal geschafft hat, bitte melden), wird hier grundsätzlich „0=Disk wurde nicht gewechselt" zurückgemeldet.

Interessant wird es erst in der Schreib-/Leseroutine ab Zeile 566. Hier erfolgt zunächst eine Überprüfung, ob das Laufwerk (jedes beliebige Laufwerk, also auch die Harddisk) softwaremäßig schreibgeschützt ist. Ist das der Fall, wird überprüft, ob auf diese Diskette geschrieben werden soll, wenn ja, wird über den Critical-Error-Handler die Meldung „Diskette ist schreibgeschützt" ausgegeben. Diese Meldung wird übrigens auch bei der RAM-Disk oder Harddisk ausgegeben. Hier ist die Bezeichnung „Diskette" vielleicht nicht ganz so zutreffend, spart aber eine Menge Speicherplatz und auch Programmierarbeit.

Falls die RAM-Disk nicht schreibgeschützt war oder nur zum Lesen zugegriffen wurde, beginnt in Zeile 596 die Berechnung der Pufferadresse. Hiernach werden die Daten aus der RAM-Disk in den gewünschten Speicherbereich verschoben. Zum Verschieben gibt es übrigens zwei Routinen: eine schnelle und (wie sollte es anders sein) eine langsamere. Die schnelle Routine wird immer dann verwendet, wenn Quell- und Zieladresse gerade sind (nur dann kann man Langwortweise auf den Speicher zugreifen), die langsamere Routine in allen anderen Fällen. Die Idee der schnellen Speicherverschiebung wurde übrigens aus dem Luftschloß [1] entnommen.

XBRA - Lister

Manchmal ist es nötig zu wissen, welche Programme sich gerade im Speicher befinden, besonders wenn diese auch noch resident sind. Deshalb wurde ST-SPEED um einen XBRA-Lister erweitert, der alle installierten Programme, die Vektoren im Bereich von $8 - $1000 verändern, ausgibt. Es werden aber nicht nur die jeweils ersten Programme innerhalb eines Vektors ausgegeben (z.B. bei den Disk-Vektoren können ein Harddisk-Treiber, eine RAM-Disk und wer weiß was noch alles installiert sein), sondern alle Programme, die sich in diesen Vektor installiert haben (sofern diese auch alle das XBRA-Protokoll unterstützen).

Die Voreinstellungen werden übrigens in einem 40 Bytes langen Bereich (Zeilen

1280 ff.) abgelegt. Dieser Bereich kann auf Disk gesichert werden und wird bei der ersten Installation automatisch eingeladen. So wird erreicht, daß z.B. Ihre Festplatte automatisch schreibgeschützt ist.

Das war der zweite Teil. Im dritten und letzten bekommen Sie die Schreibschutzroutine, den Dateischutzmechanismus, diverse Unterprogramme und die ganzen Textdaten geliefert. Bis zum nächsten Monat.

Stephan Slabihoud

Literatur:
[1] Brod/Stepper:Scheibenkleister II, MAXON Computer GmbH


```
     Fortsetzung aus ST-Computer 3/91
380: ********* RAM-Disk
381:
382: m_startram:
383:           lea       menueflag(pc),a0
384:           tst.w     (a0)
385:           bne       m_quit
386:           bsr       loesche_screen
387:           lea       installflag(pc),a0
388:           tst.w     (a0)  * Ist bereits eine
                                RAM-Disk vorhanden?
389:           bne       reinstall  * Ja, dann evtl.
                                     reinstallieren
390:
391:           lea       resident(pc),a0
392:           clr.w     (a0)       * nicht resident
393:           lea       residentchar(pc),a0
394:           move.b    #'q',(a0)
395: warte3b:  PRINT     installram(pc)
396: warte3:   bsr       inkey
397:           bset      #5,d0
398:           cmp.b     #'q',d0  * Zurück zum Menü
399:           beq       menue
400:           cmp.b     #'t',d0  * Resident <->
                                   Normal
401:           beq.s     resjump
402:           cmp.b     #'c',d0  * andere
                                   Laufwerkskennung
403:           bcc.s     char1    * BHS
404:           cmp.b     #'0',d0
405:           bcs.s     warte3   * BLO
406:           cmp.b     #'9',d0
407:           bhi.s     warte3   * BHI
408:           bsr       initram  * RAM-Disk
                                   einrichten
409:           bmi       raushier * ein Fehler ist
                                   aufgetreten
410:           lea       installflag(pc),a0
411:           move.w    #-1,(a0) * RAM-Disk
                                   installiert
412:
413: warte6:   bsr       wait
414:           bra       menue
415: char1:    cmp.b     #'p',d0  * maximal 14
                                   Laufwerke als
416:           bhi.s     warte3   * RAM-Disk möglich
417:           bclr      #5,d0
418:           lea       drivechar(pc),a0
419:           move.b    d0,(a0)
420:           sub.b     #65,d0
421:           lea       drivenr(pc),a0   * andere
                                   Laufwerkskennung
422:           move.b    d0,(a0)
423:           bra.s     warte3b
424: resjump:  lea       residentchar(pc),a0
425:           cmp.b     #'q',(a0)        * VT-52-
                                          Code (ESC q)
426:           beq.s     nowresident
427:           move.b    #'q',(a0)
428:           lea       resident(pc),a0
429:           clr.w     (a0)
430:           clr.w     d0
431:           bra.s     warte3b
432: nowresident:
433:           move.b    #'p',(a0)        * VT-52-
                                          Code (ESC p)
434:           lea       resident(pc),a0
435:           move.w    #-1,(a0)
436:           clr.w     d0
437:           bra       warte3b
438:
439: reinstall:lea       mfree(pc),a0   * RAM-Disk
                                        reinstallieren
440:           tst.l     (a0)       * Reinstallieren
                                     möglich?
441:           bne.s     reinstallok    * Ja...
442:           PRINT     impossible(pc)
443:           bsr       RAM-Disksize
444:           bra.s     warte6
445: reinstallok:
446:           PRINT     reinstallmenu(pc)
447:           bsr       RAM-Disksize    * Größe der
                                    RAM-Disk ausgeben
448: warte5:   bsr       inkey
449:           bset      #5,d0
450:           cmp.b     #'q',d0        * Zurück zum
                                          Menü
451:           beq       menue
452:           cmp.b     #'r',d0    * Reinstallieren
453:           bne.s     warte5
454:
455:           move.l    drvbits,d0
456:           clr.l     d1
457:           move.w    driveno(pc),d1
458:           tst.w     d1
459:           beq.s     exitus2
460:           bclr      d1,d0   * Laufwerk abmelden
461:           move.l    d0,drvbits
462: exitus2:  move.l    mfree(pc),-(sp)  * Speicher
                                          freigeben
463:           move.w    #$49,-(sp)        * MFREE
464:           trap      #1
465:           addq.l    #6,sp
466:           bsr       loesche_screen
467:           PRINT     reinstallram(pc)
468:           lea       installflag(pc),a0
469:           clr.w     (a0)
470:           bsr       wait
471:           bra       menue
472:
473: raushier: move.l    drvbits,d0
474:           clr.l     d1
475:           move.w    driveno(pc),d1
476:           beq.s     exitus
477:           bclr      d1,d0
478:           move.l    d0,drvbits
479: exitus:   bsr       loesche_screen
480:           PRINT     errorram(pc)    * Fehlermeld.
481:           bsr       wait
482:           bra       menue
483:
484: RAM-Disksize:
485:           PRINT     disksize(pc)    * Größe der
                                    RAM-Disk ausgeben
486:           move.l    groesse(pc),d1
487:           lea       freezahl(pc),a2
488:           bsr       binasc
489:           PRINT     freezahl(pc)
490:           rts
491:
492: initram:  ext.w     d0
493:           sub.w     #48,d0        * ASCII nach
                                        Zahl wandeln
494:           bne.s     keinenull     * gleich null?
495:           move.w    #10,d0        * ja, dann 1 MB
496: keinenull:mulu      #100,d0
497:           lea       numcl(pc),a0
498:           move.w    d0,(a0)       * Anzahl der
                                       freien Cluster
499:           add.l     #9,d0         * 9 kB
                                        Verwaltung
500:           lsl.l     #8,d0         * mal 1024
501:           lsl.l     #2,d0
502:           lea       groesse(pc),a0
503:           move.l    d0,(a0)       * Gesamtgröße
504:
505:           MALLOC    d0            * Speicher
                                        reservieren
506:           tst.l     d0
507:           beq.s     error
508:
509:           lea       puffer(pc),a0
510:           move.l    d0,(a0)       * Pufferadresse
                                        retten
511:           lea       mfree(pc),a0  * ... um die
                                          RAM-Disk
512:           move.l    d0,(a0)       * wieder zu
                                        löschen
```

→

Alles Lesens-Werte

Experimente am Schreibtisch



Durch schnelle Computer kann seit geraumer Zeit die Simulation von Naturvorgängen realistisch vorgeführt werden.
Mit diesem Buch werden Sie in die Lage versetzt, in Ihrem ATARI ST Galaxien kollidieren zu lassen, ohne daß der Himmel einstürzt, oder gar die gewagtesten chemischen Experimente zu riskieren, ohne daß gleich das Haus in die Luft gesprengt wird.
Natürlich darf bei allem Spieltrieb die Theorie der Gesetzmäßigkeiten nicht zu kurz kommen. Ausführlich dokumentierte Listings in GFA-BASIC geben einen Einblick in die programmiertechnische Seite.

Robert Becker
Computersimulationen
337 Seiten und Diskette
DM 59,-
ISBN 3-927065-03-X

Mehr als ein Buch! Mehr als nur Software!

Mittlerweile zählt Scheibenkleister II bereits zu den Standardwerken für den ST. Die Erfolgsautoren Claus Brod und Anton Stepper beschreiben auf fast 900 Seiten alles, was man über Floppies, Festplatten, CD-ROMs und andere Massenspeicher wissen muß. Das Buch beinhaltet einen großen Kursteil, in dem Themen wie Floppy- und Festplattenprogrammierung (über Betriebssystem und direkte Controllerprogrammierung), Aufzeichnungsverfahren, Anschluß von Fremdlaufwerken und sogar Justierung und Reparaturhinweise von Laufwerken u.v.m. Zusätzlich wird eine Diskette mit 1.2 MB Software mitgeliefert, die keine Wünsche offen läßt:
Track- und Sektormonitor, bootfähige RAM-Disk, Hyperformat bringt bis zu 950 kByte, Festplattentreiber CBHD.SYS (Laden von Accessories von beliebigem Laufwerk) u.v.m.



Brod/Stepper
Scheibenkleister II
872 Seiten
und Diskette
DM 89,-
ISBN 3-927065-00-5

Fürs Programmieren unentbehrlich

Die Art und Weise wie benutzererstellte Daten behandelt werden, ob beim Speichern, beim Sortieren oder beim Suchen, ist ein zentrales Thema in der Informatik. Grund genug für Dirk Brockhaus, den verschiedenen Strukturen und Algorithmen auf mehr als 400 Seiten auf den Leib zu rücken.
Angefangen mit den Grundlagen, wie einfachste Strukturelemente, über komplexe Zahlen, sowie verschiedene Methoden für Suchen und Sortieren, bis hin zur Verarbeitung großer Datenmengen umspannt dieses Buch den gesamten Themenbereich.

Dirk Brockhaus
Datenstrukturen
403 Seiten und Diskette
DM 59,-
ISBN 3-927065-02-1

Das Buch zum Handbuch

OMIKRON.BASIC, mittlerweile als ATARI-BASIC bei jedem ST beigefügt, wird in diesem Buch ausführlich beschrieben.
Eine kurze Anleitung für den Neuling führt in die grundlegenden Kennisse der BASIC-Programmierung ein.
Es folgt die Beschreibung der Befehle, Programmierkniffe mit vielen Beispielen, sowie Aufgaben und Lösungen.
Sprites werden erklärt, die Overlaytechnik zeigt das Auslagern langer Programmteile nebst Laden bei Gebrauch.
Was sind Bibliotheken (Libraries)? Das Mitbenutzen fremder Bibliotheksfunktionen, Aufrufe von TOS und GEM, sowie GDOS wird kein Geheimnis bleiben.



Ein weiterer Schwerpunkt ist die Sound- und Grafikprogrammierung unter OMIKRON-Basic, mit Klarheit über Metafiles, IMG-Format usw. Als Spezialität folgt die Beschreibung von internen Multitasking-Befehlen.

Clemens Hoffmann
OMIKRON.BASIC 3.0
355 Seiten
und Diskette
DM 59,-
ISBN 3-927065-01-3



Bestellcoupon MAXON Computer GmbH Schwalbacherstr. 52 6236 Eschborn Tel.: 06196/481811

Name/Vorname

Straße

Ort

Unterschrift

Hiermit bestelle ich:

☐ SCHEIBENKLEISTER II DM 89,00

☐ COMPUTER-SIMULATIONEN DM 59,00

☐ OMIKRON.BASIC 3.0 DM 59,00

☐ DATENSTRUKTUREN DM 59,00

Versandkosten: Inland DM 7,50
Ausland DM 10,00

Auslandbestellungen **nur** gegen Vorauskasse
Nachnahme zuzgl. DM 4,00 Nachnahmegebühr.

☐ Vorauskasse
☐ Nachnahme

Vertrieb in der Schweiz: DTZ DataTrade AG Langstrasse 94 Postfach 413 CH-8021 Zürich Tel.: 01/242 80 88 Fax.: 01/291 05 07
Vertrieb in Österreich: Dipl.-Ing. Reinhart Temmel Ges.m.b.H. & Co.KG Markt 109 A-5440 Golling Tel.: 06244/7081-17 Fax.: 06244/7188-3
Vertrieb in Frankreich: AROBACE 2 Rue Piemontesis F-75018 Paris Tel: 1/42235044 Fax 1/42545631

```
513:
514:             move.l   d0,a0          * Adresse des
                                           RAM-Disk-Puffers
515:             move.l   #18*128-1,d0   * Größe der
                                           Verwaltungssektoren
516:  clearit:   clr.l    (a0)+          * Boot-Sektor
                                           und FATs löschen
517:             dbra     d0,clearit
518:
519:             move.l   drvbits,d0     * Drvbits holen
520:             clr.w    d1
521:             move.b   drivenr(pc),d1  * Startwert,
                                           Laufwerk X-1
522:             subq.b   #1,d1
523:  fdlp:      addq.w   #1,d1          * nächstes Laufwerk
524:             cmp.w    #16,d1         * schon bei 16
                                           angekommen?
525:             beq.s    error          * ja, dann Fehler
526:             btst     d1,d0          * Laufwerk
                                           installiert?
527:             bne.s    fdlp           * ja, weitersuchen
528:
529:             bset     d1,d0          * Bit für Laufwerk
                                           setzen
530:             move.l   d0,drvbits     * Drvbits
                                           auffrischen
531:             lea      driveno(pc),a0
532:             move.w   d1,(a0)        * Laufwerks-
                                           nummer merken
533:
534:             add.w    #65,d1         * in ASCII wandeln
535:             lea      drvstr(pc),a0
536:             move.b   d1,(a0)        * in String
                                           schreiben
537:             bsr      loesche_screen
538:             PRINT    drivemsg(pc)   * Laufwerks-
                                           message
539:
540:             clr.l    d0             * kein Fehler
                                           passiert
541:             rts
542:  error:     moveq    #-1,d0
543:             rts
544:
545:  RAM-Disk_vec_reinstall:
546:             REINS_VEC hdv_rw,o_rw(pc)
547:             REINS_VEC hdv_bpb,o_bpb(pc)
548:             REINS_VEC hdv_mediach,o_media(pc)
549:             rts
550:
551:             dc.b     'XBRA'
552:             dc.l     kennung
553:  o_bpb:     dc.l     0
554:  mybpb:     lea      installflag(pc),a0
555:             tst.w    (a0)
556:             beq.s    mybpb1         * keine RAM-Disk
vorhanden
557:             move.w   driveno(pc),d0
558:             cmp.w    4(sp),d0
559:             beq.s    newbpb
560:  mybpb1:    move.l   o_bpb(pc),a0
561:             jmp      (a0)
562:  newbpb:    lea      protobpb(pc),a0
563:             move.l   a0,d0
564:             rts
565:
566:             dc.b     'XBRA'
567:             dc.l     kennung
568:  o_rw:      dc.l     0
569:  myrwabs:   move.w   14(sp),d0      * Disknummer
570:             move.l   protect(pc),d1
571:             btst     d0,d1          * Ist Bit
                                           gesetzt?
572:             bne.s    no_write       * Ja, dann nicht
                                           schreiben...
573:  is_read:   lea      installflag(pc),a0
574:             tst.w    (a0)
575:             beq.s    myrwabs1       * keine RAM-Disk
                                           vorhanden
576:             move.w   driveno(pc),d0
577:             cmp.w    14(sp),d0
578:             beq.s    newrw
579:  myrwabs1:  move.l   o_rw(pc),a0
580:             jmp      (a0)
581:  no_write:  move.w   4(sp),d0
582:             btst     #0,d0          * Lesen oder
                                           schreiben?
583:             beq.s    is_read        * Lesen, dann
                                           nichts machen...
584:  retry:     move.l   #-13,d0        * Schreiben ->
                                           Fehler...
585:             move.w   14(sp),-(sp)   * Disknummer
586:             move.w   d0,-(sp)       * Error
587:             moveq    #-1,d0
588:             move.l   etv_critic,a0  * ETV_CRITIC
589:             jsr      (a0)           * Toll! Eine
                                           schreibgeschützte
590:             addq.l   #4,sp          * RAM-Disk
591:             cmp.l    #$10000,d1     * -XX=Abbruch,
                                           $10000=Retry
592:             beq.s    retry
593:             move.l   #-13,d0
594:             rts
595:
596:  newrw:     clr.l    d0             * d0.l vorbereiten
597:             move.w   12(sp),d0      * Sektoranzahl
598:             lsl.l    #8,d0
599:             lsl.l    #1,d0
600:             move.l   6(sp),a0       * Pufferadresse
601:             move.w   10(sp),d1      * Sektoranzahl
602:
603:             move.l   puffer(pc),a1
604:             add.l    d0,a1          * plus Offset
605:             move.w   4(sp),d0       * R/W-Flag
606:             btst     #0,d0          * Lesen oder
                                           schreiben?
607:             beq.s    copysec        * Lesen
608:             exg      a0,a1          * zum Schreiben
                          nur Pufferadresse tauschen
609:
610:  copysec:   move.l   a0,d0          * Adresse gerade?
611:             btst     #0,d0          * Bit 0 gesetzt?
612:             bne.s    ungerade       * ja
613:             move.l   a1,d0          * Adresse gerade?
614:             btst     #0,d0
615:             beq.s    gerade         * nein
616:
617:  ungerade:  subq.l   #1,d1
618:  ram_1:     moveq.l  #63,d0
619:  ram_lp:    move.b   (a1)+,(a0)+
620:             move.b   (a1)+,(a0)+
621:             move.b   (a1)+,(a0)+
622:             move.b   (a1)+,(a0)+
623:             move.b   (a1)+,(a0)+
624:             move.b   (a1)+,(a0)+
625:             move.b   (a1)+,(a0)+
626:             move.b   (a1)+,(a0)+
627:             dbra     d0,ram_lp
628:             dbra     d1,ram_1
629:             bra.s    newmedia
630:  gerade:    movem.l  d2-d7/a2-a6,-(sp)
631:             move.w   d1,-(sp)
632:  ram_even:  movem.l  (a1)+,d0-d7/a2-a6
                  * Diese superschnelle Kopierroutine
633:             movem.l  d0-d7/a2-a6,(a0)
                  * geht auf's Luftschloß
634:             movem.l  (a1)+,d0-d7/a2-a6
                  * (Scheibenkleister) zurück.
635:             movem.l  d0-d7/a2-a6,52(a0)
                  * Ist über 13.5 MBit/s schnell..
636:             movem.l  (a1)+,d0-d7/a2-a6
637:             movem.l  d0-d7/a2-a6,104(a0)
638:             movem.l  (a1)+,d0-d7/a2-a6
639:             movem.l  d0-d7/a2-a6,156(a0)
640:             movem.l  (a1)+,d0-d7/a2-a6
641:             movem.l  d0-d7/a2-a6,208(a0)
642:             movem.l  (a1)+,d0-d7/a2-a6
643:             movem.l  d0-d7/a2-a6,260(a0)
644:             movem.l  (a1)+,d0-d7/a2-a6
645:             movem.l  d0-d7/a2-a6,312(a0)
646:             movem.l  (a1)+,d0-d7/a2-a6
647:             movem.l  d0-d7/a2-a6,364(a0)
648:             movem.l  (a1)+,d0-d7/a2-a6
649:             movem.l  d0-d7/a2-a6,416(a0)
650:             movem.l  (a1)+,d0-d7/a2-a4
651:             movem.l  d0-d7/a2-a4,468(a0)
652:             lea      512(a0),a0
653:             subq.w   #1,(sp)
654:             bne.s    ram_even
655:             tst.w    (sp)+
656:             movem.l  (sp)+,d2-d7/a2-a6
657:  newmedia:  moveq.l  #0,d0
658:             rts
659:
```

→

```
660:            dc.b       'XBRA'
661:            dc.l       kennung
662: o_media:   dc.l       0
663: mymedia:   lea        installflag(pc),a0
664:            tst.w      (a0)
665:            beq.s      mymedia1    * keine RAM-Disk
                                         vorhanden
666:            move.w     driveno(pc),d0
667:            cmp.w      4(sp),d0
668:            beq.s      newmediax
669: mymedia1:  move.l     o_media(pc),a0
670:            jmp        (a0)
671: newmediax:clr.l       d0
672:            rts
673:

763: ********* SAVE - Default
764:
765: m_save:    bsr        file_reinstall * Schutz
                                         deaktivieren
766:            bsr.s      m_save2
767:            lea        fileflag(pc),a0
768:            tst.w      (a0)
769:            beq        menue           * Schutz
                                 nicht mehr aktivieren
770:            bsr        file_install  * Schutz
                                         wieder an
771:            bra        menue
772: m_save2:   FCREATE    infoname(pc)  * Datei
                                         erstellen
773:            tst.w      d0
774:            bmi.s      serror
775:            move.w     d0,d7
776:            FWRITE     save_area(pc),#40,d7
                                    * Daten schreiben
777:            FCLOSE     d7       * Datei schließen
778: serror:    rts
779:
780: ********* LOAD - Default
781:
782: m_load:    bsr.s      m_load2
783:            bra        menue
784: m_load2:   FOPEN      infoname(pc) * Datei öffnen
785:            tst.w      d0
786:            bmi.s      lerror
787:            move.w     d0,d7
788:            FREAD      save_area(pc),#40,d7
                                    * Daten laden
789:            FCLOSE     d7       * Datei schließen
790: lerror:    rts
791:
792: ********* XBRA - List
793:
794: m_xbra:    lea        menueflag(pc),a0
795:            tst.w      (a0)
796:            bne        m_quit
797:            bsr        loesche_screen
798:            PRINT      xbra_home(pc)
799:            clr.l      d6        * Zeilenzähler = 0
800:
801:            move.w     #8,a6     * a6 - Vector
802: xbra_a:    move.l     (a6),d7    * d7 - Start
803:            btst       #0,d7      * Gerade Adresse
804:            bne        xbra_next  * Nein!
805:            cmp.l      #12,d7     * Größergleich 12
806:            bcs        xbra_next  * Lower (LO)
807:            cmp.l      #$3ffffc,d7  * <= $3ffffe
808:            bhi        xbra_next  * Größer
809: xbra_rep:  move.l     d7,a5
810:            cmp.l      #'XBRA',-12(a5) * XBRA?
811:            bne        xbra_next     * kein XBRA
812:
813:            lea        xbra_kennung(pc),a2
814:            move.l     -8(a5),(a2)    * Kennung
                                          eintragen
815:
816:            lea        xbra_vector(pc),a2
                                  * Vector-Adresse
817:            move.l     a6,d1      * Wert
818:            moveq.l    #4,d0      * Länge
819:            bsr        binhex
820:
821:            lea        xbra_adresse(pc),a2
                                  * Adresse
822:            move.l     d7,d1      * Start
823:            moveq.l    #8,d0
824:            bsr        binhex
825:
826:            lea        xbra_old(pc),a2  * Adresse
827:            move.l     -4(a5),d1        * Old
828:            moveq.l    #8,d0
829:            bsr        binhex
830:
831:            movem.l    d0-d7/a0-a6,-(sp)
832:            PRINT      xbra_text(pc)    * Ausgeben
833:            movem.l    (sp)+,d0-d7/a0-a6
834:            addq.l     #1,d6            * Zeile + 1
835:            cmp.l      #10,d6           * 10 Zeilen
                                          ausgegeben?
836:            bcs.s      xbra_max * Nein! (LO)
837:            clr.l      d6       * Zeilenzähler = 0
838:            bsr        wait
839:            bsr        loesche_screen
840:            movem.l    d0-d7/a0-a6,-(sp)
841:            PRINT      xbra_home(pc)
842:            movem.l    (sp)+,d0-d7/a0-a6
843: xbra_max:  move.l     -4(a5),d7     * Neuer Start
844:            btst       #0,d7         * ungerade?
845:            bne.s      xbra_next     * Ja...
846:            cmp.l      #12,d7        * < 12
847:            bcs.s      xbra_next     * Ja (LO)
848:            cmp.l      #$3ffffc,d7   * > $3ffffe
849:            bhi.s      xbra_next     * Ja
850:            bra        xbra_rep     * gleicher
                                         Vector...
851: xbra_next:addq.l      #2,a6
852:            cmp.l      #$1000,a6     * Nur bis
                                         Adresse $1000
853:            bls        xbra_a
854:
855:            bsr        wait
856:            bra        menue
```

Liebling, ich habe die Kinder geschrumpft



Verbesserung des LZW-Komprimierverfahrens

In der ST-Ecke im Maiheft letzten Jahres wurde der LZW-Algorithmus beschrieben, der etwa in den Programmen ARC und ZOO Verwendung findet. Ausführlich wurde die optimierte Suche mittels Hash-Algorithmus beschrieben.

Als Ergänzung soll hier der eigentliche LZW-Algorithmus etwas detaillierter beleuchtet werden. Dabei ergeben sich zwei Verbesserungen; insbesondere läßt sich die Kompression gleichzeitig beschleunigen und vereinfachen, da keinerlei Suche mehr erforderlich ist und somit auch keine Suchbeschleunigung.

Prinzip

Die Bytes der Eingabedatei werden gruppiert zu Byte-Sequenzen von einem oder mehreren Byte. In jedem Kompressionsschritt erfolgt die Gruppierung zu einer solchen Byte-Sequenz, und als Repräsentant für die Sequenz wird ein 10-Bit-Codewort in die Ausgabedatei geschrieben. Ist bei der Dekompression die Zuordnung Codewort-Byte-Sequenz bekannt, läßt sich aus den 10-Bit-Codeworten die Ursprungsdatei rekonstruieren. Um keinen Platz zu verschenken, dürfen in der Ausgabedatei die 10-Bit-Worte nicht in je 2 Bytes eingebettet werden.

Der Kompressionseffekt kommt dadurch zustande, daß, wenn die Byte-Sequenz der Eingabedatei aus zwei oder mehr Bytes besteht, 16 oder mehr Bits der Eingabedatei durch 10 Bits der Ausgabedatei repräsentiert werden. Umgekehrt erfährt eine Byte-Sequenz aus einem einzelnen Byte allerdings eine Aufblähung von 8 auf 10 Bits. Das ist auch der Grund für die Beschränkung auf 10 Bits; bei einer Codierung mit mehr Bits wären diese Kompressionsverhältnisse ungünstiger. Bei nur 9 Bits Codebreite wären sie hingegen günstiger; es könnten aber auch nur 512 verschiedene Byte-Sequenzen dargestellt werden, recht wenig angesichts der allein 256 möglichen 1-Byte-Sequenzen.

Byte-Sequenzen und ihre Codierung

Wir starten mit dem Vorrat der 256 1-Byte-Sequenzen als Schablonen für die zu gruppierenden Bytes der Eingabedatei. Als Code wählen wir die 10-Bit-Erweiterung des jeweiligen Bytes.

In jedem Schritt der Kompression werden wir eine neue Schablone bilden und ihr einen Code zuordnen, so daß der Vorrat an Schablonen ständig wächst.

Ein Kompressionsschritt sieht folgendermaßen aus: in der Abfolge der nächsten Bytes der Eingabedatei wird die längste Schablone gesucht, die mit dieser Abfolge übereinstimmt. So ergibt sich die nächste Byte-Gruppierung, und der Code der damit identischen Schablone wird in die Ausgabedatei geschrieben.

Diese Schablone, ergänzt um das nächste Byte der Eingabedatei, wird neue Schablone dieses Schritts. Als zugehörigen Code vergeben wir im ersten Kompressionsschritt 256 (weil die Werte 0...255 schon belegt sind für die Ausgangsschablonen), im zweiten 257 usw., d.h. eine laufende Nummer.

Aufgrund dieser Vorgehensweise ist sichergestellt, daß in jedem Kompressionsschritt

- eine *neue* Schablone bereitgestellt wird
- eine Schablone mit mehr als 2 Bytes nur dann gebildet wird, wenn die um das letzte Byte verringerte Teilsequenz sich bereits bewährt hat, d.h. mindestens zweimal in der Eingabedatei vorgekommen ist. Dies verhindert zumindest tendenziell exotische, lange Schablonen, läßt aber durchaus umfangreiche Schablonen zu, wie sie etwa in Text-Files zur Darstellung von wiederholten Blanks willkommen sind.

- der Code, der in die Ausgabedatei geschrieben wird, < 255+n ist im n-ten Kompressionsschritt. Das bedeutet, daß eine ganze Weile nur Codes < 512 ausgegeben werden, und diese sind mit 9 Bits darstellbar. Man verwendet daher variable Codebreite und schreibt zunächst mit 9 Bits Breite. Sobald ein Code >= 511 zu schreiben ist, schreibt man zunächst 511 als Steuercode, der besagt, daß die folgenden Codeworte mit 10 Bits Breite ausgegeben werden, und dann den Code (und alle folgenden) mit 10 Bits.

Da in jedem Kompressionsschritt eine neue Schablone gebildet wird, zu deren Codierung aber nur 10 Bits zur Verfügung stehen, bricht das Verfahren nach spätestens 1024 Kompressionsschritten ab. Es werden dann alle Schablonen gelöscht (bis auf die 256 Ausgangsschablonen), eine entsprechende Information in die Ausgabedatei geschrieben (Steuercode 1023) und eine neue Kompressionsphase eingeleitet mit neu aufzubauenden Schablonen. [Eine Variante bildet weiter neue Schablonen (mit 11-Bit-Codierung) und bricht erst ab, wenn ein solcher 11-Bit-Code in die Ausgabedatei zu schreiben wäre.]

Eine Kompressionsphase besteht also aus 767 Schritten (in denen die Codes 256...1022 als Schablonencodes vergeben werden). 255 Codes und 1 Steuercode werden dabei mit 9 Bits, und 512 Codes und 1 Steuercode mit 10 Bits ausgegeben; insgesamt beträgt das Ausgabevolumen 7434 Bits. Bei einer Kompressionsrate von ca. 50% (ziemlich typisch) umfaßt eine Kompressionsphase somit ca. 15000 Bits in der Eingabedatei, d.h. rund 2 kB.

Dies hat insofern Vorteile, als die Bildung der Schablonen in Bereichen von je nur ca. 2 kB erfolgt, so daß eine laufende Anpassung an die Gegebenheiten der zu komprimierenden Datei erfolgt.

Dekompression

Im Dekompressionsschritt Nr. n wird

a) das n-te Codewort der komprimierten Datei dekodiert und das Ergebnis in die Ausgabedatei geschrieben. Ist der Code <= 255, handelt es sich um ein Byte mit dieser Wertigkeit, ansonsten um den Code einer bereits bekannten Mehr-Byte-Schablone.

b) das Ergebnis aus a) als die zu Code 255+n gehörige Schablone bestimmt - es fehlt allerdings noch das letzte Byte.

c) das 1. Byte des Ergebnisses aus a) als das fehlende Byte des letzten Dekompressionsschritts abgestellt [als das letzte Byte der zu Code 255+(n-1) gehörenden Schablone].

```
c := nächstes Codewort (erweitert auf 16 Bits)          [Teil a)]
     CODE_ADR[n] := aktuelle Adresse im Ausgabepuffer       [Teil b)]
     c <= 255 ? Ja:    CODE_LEN[n] := 2                     [Teil b)]
                       Übertrage c als Byte in den Ausgabepuffer
                                                            [Teil a),c)]
               Nein:   CODE_LEN[n] := CODE_LEN[c] + 1       [Teil b)]
                       Übertrage die Byte-Sequenz ab CODE_ADR[c] der Länge
                       CODE_LEN[c] von links nach rechts in den
                       Ausgabepuffer                     [Teil a),c)]
```

Tab. 1: Algorithmus für Schritt n (ohne Steuercode-Feinheiten)

```
b := nächstes Byte der Originaldatei
               c := b (als 16-Bit-Wert)
     Iteriere: b := nächstes Byte der Originaldatei
               DAUGHTER[c,b] = NO ?   Ja :  Schreibe c als 10-Bit-Wert in
                                            die Ausgabedatei
                                            DAUGHTER[c,b] := 255+n
                                            Beende die Iteration
                                      Nein: c := DAUGHTER[c,b]
                                            Iteriere weiter
```

Tab. 2: Algorithmus für Kompressionsschritt n (es gelte NO=-1)

Bei der Implementierung muß man die Punkte a) und c) gleichzeitig behandeln, sonst läßt sich a) nicht durchführen im Falle, daß das n-te Codewort den Wert 255+(n-1) trägt.

Am einfachsten läßt sich das wohl so implementieren: Die Ergebnisse aus a) werden in einem Ausgabepuffer abgestellt. Darüber hinaus wird eine Tabelle aufgebaut mit dem Code als Index und Anfangsadresse und Länge der zugehörigen Byte-Sequenz im Ausgabepuffer (!) als Tabelleneinträge (s. Tab 1). Es wird also keine getrennte Code-Tabelle aufgebaut, denn die Struktur der Zerlegung der Originaldatei bildet selbst implizit die Code-Tabelle.

Kompression

Der wesentliche Punkt in jedem Kompressionsschritt ist: wie findet man die längste übereinstimmende Schablone? Dazu bildet man iterativ aus der Abfolge der nächsten Bytes der Eingabedatei solange eine sich um je 1 Byte verlängernde Teilsequenz, wie es eine Schablone gibt, die mit dieser Teilsequenz übereinstimmt.

Die Prüfung, ob es eine mit der Teilsequenz übereinstimmende Schablone gibt, kann ohne aufwendigen Vergleich erfolgen, ja man braucht sich die Schablonen nicht einmal zu merken. Dazu muß man lediglich beachten, daß nach Konstruktion jede Schablone mit mehr als 1 Byte zustandekommt als Tochter aus der Vereinigung einer Mutter-Schablone und einem Vater-Byte. Diese Beziehung läßt sich kompakt darstellen, wenn man Mutter, Vater und Tochter durch ihren Code repräsentiert. Dementsprechend gibt es neben dem Ein- und Ausgabepuffer eine Tabelle mit den Mutter-Vater-Tochter-Beziehungen.

DAUGHTER[c,b] gibt an, ob die zu Code *c* gehörige Schablone, ergänzt um Byte *b*, bereits Schablone ist oder nicht. *DAUGHTER[c,b]* enthält einen 16-Bit-Integer-Wert, und zwar -1, falls die Sequenz aus *c* und *b* keine Schablone ist, und ansonsten den Code der Tochter-Schablone von *c* und *b* (s. Tab. 2).

Und wo ist der Haken? Die DAUGHTER-Tabelle ist ziemlich groß: 1024*256 Tabelleneinträge à 16 Bits, das macht 512 kB. Für die meisten ST-Besitzer dürfte das jedoch kein Problem sein; STs mit weniger als 1 MB sind angesichts der mäßigen Aufrüstungskosten wohl nur noch selten anzutreffen. Ein wenig Beachtung verdient die Initialisierung einer derart großen Tabelle. Zu Beginn der Kompression muß die ganze Tabelle auf den NO-Wert gesetzt werden mit optimierten Befehlen. Nach jeder Kompressionsphase reicht es, die belegten Plätze zurückzusetzen (767 Stück). Dazu muß zusätzlich eine Tabelle mit den belegten Adressen der DAUGHTER-Tabelle geführt werden.

Weitere Möglichkeiten

Läßt man höheren Aufwand zu, ergeben sich neue Möglichkeiten. Zum einen kann man den Platzbedarf der DAUGHTER-Tabelle stark reduzieren, wenn man mit 2 Tabellen arbeitet: *HI_DAUGHTER* und *LO_DAUGHTER* mit jeweils nur 1024*16 Einträgen.

Dazu betrachten wir das jeweilige Vater-Byte *b* und splitten es auf in oberes Halb-Byte *h* und unteres Halb-Byte *l*.

HI_DAUGHTER[c,h] gibt an, ob die zu Code *c* gehörige Schablone, ergänzt um Halbbyte *h*, zusammen mit irgendeinem weiteren Halb-Byte bereits Schablone ist oder nicht. Falls nicht, gilt *HI_DAUGHTER[c,h]=NO*, ansonsten enthält *HI_DAUGHTER[c,h]* einen Index *i* auf die Tabelle *LO_DAUGHTER*, und *LO_DAUGHTER[i,l]* gibt an, ob die aus *c*, *h*

und *l* gebildete Sequenz bereits Schablone ist oder nicht (falls ja, enthält *LO_-DAUGHTER[i,l]* den Code dieser Schablone, ansonsten *NO*). Auf diese Weise benötigen wir nur noch 2 Tabellen mit 1024*16 Einträgen à 16 Bits, insgesamt also 64 kB.

Denkt man an Text-Files, wo nur größenordnungsmäßig 60 verschiedene Bytes vorkommen, sind rund 200 Codeworte unnütz belegt mit nie vorkommenden 1-Byte-Sequenzen. Die Kompressionsphase könnte rund 20% länger sein, was besonders interessant ist, weil am Ende der Phase die meisten Schablonen zur Verfügung stehen und sogar noch ca. 200 weitere Schablonen hinzukommen könnten. In diesem Teil der Kompressionsphase bestehen also besonders gute Chancen auf Repräsentation einer Mehr-Byte-Sequenz durch 1 Codewort.

Die Lösung für diese Problematik besteht darin, die 1-Byte-Schablonen nicht von vorneherein zu codieren, sondern erst dann, wenn sie aktuell in der Eingabedatei vorkommen.

Zu realisieren ist das recht einfach: Ist eine bis dato unbekannte 1-Byte-Sequenz zu codieren, geschieht dies anstelle (!) des vorgegegebenen Codes durch einen Steuercode (in der aktuellen Codebreite), gefolgt von Informationen über das zu definierende Byte. Der Steuercode könnte zum Beispiel 0 sein, gefolgt vom Byte selbst. Gegenüber dem ursprünglichen Verfahren sind dann aber bei jeder neuen 1-Byte-Schablone 8 Bits zusätzlich auszugeben.

Günstiger ist es, die Codes 0...15 als Steuercode für neue 1-Byte-Schablonen zu benutzen und sie gleichzeitig die Information über das obere Halb-Byte des zu definierenden Bytes tragen zu lassen. Dann brauchen nur noch 4 Bits zu folgen, die das untere Halb-Byte beschreiben. Diese Variante besitzt den Vorteil, daß jetzt das Schreiben in die Ausgabedatei bereits mit 5 Bits Codebreite beginnen kann, denn nur noch die Codes 0...15 sind von Anfang an belegt.

Wie sieht das Ausgabevolumen aus? Um hiervon einen Eindruck zu bekommen, gehen wir von der pessimistischen (aber dennoch realistischen) Annahme aus, daß alle 1-Byte-Schablonen zu Beginn der Kompressionsphase definiert werden. Pessimistisch ist die Annahme deshalb, weil in einem Schritt mit einer neuen 1-Byte-Sequenz gleich 2 Codes vergeben werden, und deshalb die jeweilige Umschaltschwelle auf nächsthöhere Codebreite früh erreicht wird. Ferner werden wir vereinfachend davon ausgehen, daß bei Erreichen der Umschaltschwelle durch die vergebenen (nicht die ausgegebenen) Codes die Ausgabe mit nächsthöherer Codebreite erfolgt - auch eine pessimistische Annahme, da die Codebreitenumschaltung nicht ohne Anlaß erfolgt. Dafür wollen wir die Umschaltcodes und den Ende-Steuercode vernachlässigen. Wir arbeiten so mit einem eher pessimistischen, aber einigermaßen realistischen und einfachen Modell.

```
Kompressionsschritt n:
          c := 0
Iteriere: b := nächstes Byte der Eingabedatei
          DAUGHTER(c,b) = NO ?  Ja:   Schreibe c in die Ausgabedatei
                                      Beende die Iteration
                                Nein: c := DAUGHTER(c,b)
                                      Iteriere weiter

Funktion DAUGHTER(c,b):
lokale Variablen:  h,l,i,cd
globale Variablen: lastcode, nextindex
h := oberes Halb-Byte aus b
l := unteres Halb-Byte aus b
i := HI_DAUGHTER[c,h]
i = NO ?    Ja:    HI_DAUGHTER[c,h] := nextindex
                   i := nextindex
                   nextindex:= nextindex + 1
cd := LO_DAUGHTER[i,l]
cd = NO ?   Ja:    lastcode := lastcode + 1
                   LO_DAUGHTER[i,l] := lastcode
                   c = 0 ?   Ja:   Gebe h in aktuellller Codebreite aus,
                                   gefolgt von l in 4 Bits Breite.
                                   Verlasse die Funktion mit dem Wert
                                   lastcode
                             Nein: Verlasse die Funktion mit dem Wert
                                   NO
            Nein: Verlasse die Funktion mit dem Wert cd
Dekompressionsschritt n (lastcode ist zu Beginn der Dekompression mit
                         15 vorbesetzt):
lastcode := lastcode + 1
c := nächstes Codewort
CODE_ADR[lastcode] := aktuelle Adresse im Ausgabepuffer
c <= 15 ?   Ja:    l := nächstes Codewort in 4 Bits Breite
                   CODE_LEN[lastcode] := 1
                   lastcode := lastcode + 1
                   CODE_ADR[lastcode] := aktuelle Adresse im
                                         Ausgabepuffer
                   CODE_LEN[lastcode] := 2
                   Übertrage c und l als je ein Halb-Byte
                   in den Ausgabepuffer
            Nein:  CODE_LEN[lastcode] := CODE_LEN[c] + 1
                   Übertrage die Byte-Sequenz ab CODE_ADR[c] der Länge
                   CODE_LEN[c] in den Ausgabepuffer
```

Tab. 3: Kompressionsschritt n (nextindex ist zu Beginn der Kompression mit 0 vorbesetzt, lastcode mit 15 und alle Tabelleneinträge mit NO=-1)

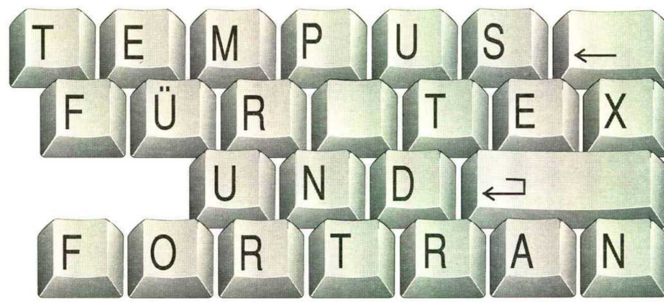
Im ersten Kompressionsschritt werden dann die Codes 16,17 und im achten Schritt die Codes 30,31 vergeben. In den ersten 8 Schritten wird daher mit 5 Bits Breite ausgegeben, analog in den nächsten 16 Schritten mit 6 Bits Breite usw.

Wie sehen nun die Verhältnisse aus bei Text-Files mit 60 verschiedenen 1-Byte-Schablonen? In den ersten 60 Kompressionsschritten werden alle 1-Byte-Schablonen definiert; ausgegeben werden 8*(5+4)+16*(6+4)+32*(7+4)+4*(8+4)=632 Bits. Zum Vergleich mit dem Originalverfahren wollen wir wie dort von 767 Kompressionsschritten ausgehen (obwohl jetzt 256-60-16=180 Schritte mehr möglich sind). Es sind also noch 707 Schritte zu berücksichtigen. Davon wird in 128-4=124 Schritten mit 8 Bits ausgegeben, in 256 Schritten mit 9 Bits und in den verbleibenden 327 Schritten mit 10 Bits. Insgesamt werden 7198 Bits ausgegeben, geringfügig weniger als die 7434 Bits des Originalverfahrens.

Wie sieht es aus im ungünstigen Fall von annähernd 256 verschiedenen 1-Byte-Schablonen? Wir wollen von 240 ausgehen, da andernfalls gar keine 767 Kompressionsschritte zustandekommen. Die entsprechende Rechnung ergibt dann 8166 Bit, also rund 10% mehr als beim Originalverfahren. Die Kompressionsrate verringert sich um 5% (bei einer typischen Kompressionsrate von 50%).

Mit 5% Einbuße bei der Kompressionsrate muß also schlimmstenfalls gerechnet werden. Inwieweit es in praxi überhaupt zu einer Einbuße kommt, ist offen. Insgesamt kann man die Variante als Universalverfahren ansehen, das für die Komprimierung von Text-Files optimiert ist. Der Klarheit wegen sei der Kompressions- und Dekompressions-Algorithmus für eine Kombination beider Varianten in Tab. 3 aufgeführt.

Horst Albrecht



Tastaturanpassung für zwei typische Anwendungen.

Wie man einen Editor optimal an die eigene Arbeitsweise und das gewünschte Einsatzgebiet anpaßt, zeigt dieser Artikel am Beispiel des wegen seiner Geschwindigkeit beliebten und weit verbreiteten Tempus. Aus den hier angebotenen drei Varianten kann man sich leicht eine persönliche Version kombinieren und sie durch eigene Ergänzungen abrunden.

TEMPUS ist ein Hochgeschwindigkeits-Editor für die Eingabe von Texten im professionellen Bereich. Typische Einsatzgebiete sind Quelltexte in verschiedenen Programmiersprachen und für das Satzsystem TeX. Völlig ungeeignet ist TEMPUS dagegen zur WYSIWYG-Formatierung (What You See Is What You Get) ausgabefertiger Texte, dafür gibt es genügend andere Programme, nicht zuletzt TEMPUS-WORD aus dem gleichen Hause.

TEMPUS findet also immer dann Verwendung, wenn das Ergebnis reiner Text sein soll, der keinerlei Steuerzeichen enthält und anschließend von einem anderen Programm weiterverarbeitet wird. Zusätzliche Informationen, wie etwa Zeilenlineale und unterschiedliche Zeichensätze, würden hier nur stören. Für diesen Einsatz dürfte TEMPUS aber auch das am häufigsten verwendete Programm sein; EDISON ist ja noch nicht lange am Markt, und das PD-Programm MicroEMACS hat wohl vor allem wegen der fehlenden Dateiauswahlboxen auf dem Atari leider keinen allzu großen Freundeskreis.

Der angesprochene Einsatzschwerpunkt ist nun einerseits gekennzeichnet durch immer wiederkehrende Zeichenfolgen, wie etwa Befehle in TeX und Schlüsselwörter in den Programmiersprachen, andererseits sind es gerade diese Anwender gewohnt, zu programmieren - in einer Programmiersprache oder in TeX - und somit geneigt, die in TEMPUS gebotenen Möglichkeiten auszunutzen, um sich, wenn schon nicht das Leben, so doch wenigstens die Eingabe zu erleichtern. Leider bietet nun TEMPUS nicht die weitgehenden Programmiermöglichkeiten wie etwa der MicroEMACS, aber mit Hilfe der Tastenbelegung über die INS-Datei und einer geeigneten Einstellung der Grundparameter kommt man doch recht weit. Dies wollen wir in dem vorliegenden Artikel exemplarisch an zwei typischen Einsatzgebieten, TeX und FORTRAN, vorführen. Diese beiden Bereiche sind insofern besonders reizvoll, da sie für einen Editor ganz unterschiedliche Aufgaben bereithalten. Einmal geht es um die flüssige Eingabe von Texten, wobei der Anwender nur die Wörter eingeben möchte und der Editor alle anderen Aufgaben, wie etwa den Zeilenumbruch, zu übernehmen hat, zum anderen um die streng reglementierte Anordnung von Anweisungsteilen in vorgegebene Bereiche einer Zeile.

Die Grundversion

Zunächst werden wir jedoch eine für alle Anwendungen sinnvolle Grundversion entwickeln, die mit den weitergehenden Anwendungen dann kombiniert werden kann. Dabei erschließen wir die elementaren Zeigerbewegungen und einige Sonderzeichen für das 10-Finger-System.

Nicht alle Programmierer sind Zwei-Finger-Brutalos, und gerade TeX-Texte werden häufig von Leuten eingegeben, die das Schreiben mit allen zehn Fingern beherrschen. Diese Fähigkeit bildet das hauptsächliche Kapital eines Vielschreibers; Editor oder Textprogramm sollten daher diese Arbeitsweise so weit wie möglich unterstützen. In der Werbung für das Textprogramm Wordstar unter MS-DOS wird dies im übrigen besonders hervorgehoben. Neben der reinen Texteingabe am häufigsten sind einfache Zeigerbewegungen (um ein Zeichen/Wort nach links/rechts, um eine Zeile/Bildschirmseite nach oben/unten) und das Löschen einzelner Zeichen mit Backspace oder Delete. Während die Tasten Backspace und Delete zur Not noch mit dem kleinen Finger der rechten Hand zu erreichen sind, muß bei vielen Programmen für Zeigerbewegungen die rechte Hand aus der Grundstellung genommen werden, um die Pfeiltasten oder gar die Maus zu bedienen. Dies bremst den Schreibfluß in ärgerlicher Weise.

Es ist völlig unverständlich, warum man einerseits Unsummen für die Ergonomie am Arbeitsplatz ausgibt, andererseits jedoch solche einfachen und sinnvollen Maßnahmen, die nicht einmal etwas kosten würden, vollständig mit Verachtung straft. Man könnte fast auf den Verdacht kommen, daß Sicherheitsingenieure und Gewerkschaftler grundsätzlich nicht mit zehn Fingern schreiben können und Sekretärinnen erst gar nicht gefragt werden.

Das Vorbild

Manche kennen noch das oben bereits erwähnte Programm Wordstar (etwa unter CP/M oder MS-DOS), bei dem die elementaren Zeigerbewegungen über den Umschalter Control aus der Zehn-Finger-Haltung heraus verfügbar sind. Nun kann

man unter TEMPUS ja alle Tastenkombinationen frei belegen. Dies nutzen wir aus, um die Wordstar-Befehle zur Zeigerbewegung nachzubilden. Zusätzlich legen wir noch den Rückstrich (backslash) und die geschweiften Klammern auf Tastenkombinationen, die ebenfalls leicht im 10-Finger-System zu greifen sind. Geschweifte Klammern findet man in manchen Programmiersprachen, in TeX sind diese Klammern und der Rückstrich ebenfalls sehr beliebt.

Bild 1 zeigt die neue Belegung der Tasten; wie man sieht, mußten drei Tempus-Funktionen der Wordstar-Kompatibilität weichen. Zwei davon wurden kurzerhand auf andere Tastenkombinationen umquartiert, die Funktion *Text mit Duplikat speichern* kann zukünftig nur noch über das Menü abgerufen werden. Dies spiegelt die persönlichen Vorlieben des Autors, der lieber mit der Funktion *Text ohne Duplikat speichern* (Alternate,[S]) arbeitet. Bei Bedarf legt man auf diese Tastenkombination einfach das Speichern MIT Duplikat. Zusammenfassend zeigt Bild 2 die Wordstar-kompatiblen Zeigerbewegungen, die jetzt möglich sind.

Umschalter,Taste	Neue Bedeutung
Control,[ö]	\
Control,[(]	{
Control,[)]	}
Control,[E]	Zeiger nach oben (früher: Ersetzen)
Control,[S]	Zeiger nach links (früher: Text mit Duplikat speichern)
Control,[D]	Zeiger nach rechts
Control,[X]	Zeiger nach unten (früher: Ende mit Sicherheitsabfrage)
Alternate,[E]	Ersetzen
Shift,Alternate,[X]	Beenden mit Sicherheitsabfrage

Bild 1: Geänderte Tastenbelegung der Grundversion

<Control> & ...	Zeiger um ein(e)...
A	Wort nach links
S	Zeichen nach links
D	Zeichen nach rechts
F	Wort nach rechts
R	Seite nach oben
E	Zeile nach oben
X	Zeile nach unten
C	Seite nach unten

Bild 2: Wordstar-kompatible Zeigerbewegungen

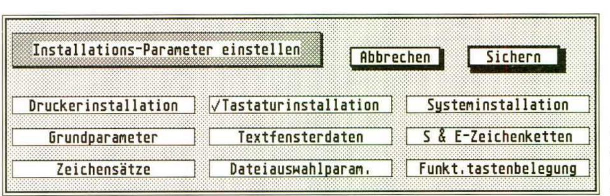


Bild 3: Zu speichernde Parametergruppe auswählen.

Tastenbelegung anpassen

Die Tastaturinstallation ändert man in Tempus mit Hilfe sogenannter Quelldaten, die in einer Installationsdatei (Endung meist: „.INS") abgelegt werden. Ein Beispiel einer solchen Installationsdatei ist das auf der Originaldiskette beigefügte QUELLDAT.INS, in der die Standardbelegung definiert wird. Man benötigt nun lediglich eine solche Datei und zwei Tempus-Funktionen, um eine beliebige Änderung der Tastaturbelegung zu bewerkstelligen. Dies erreicht man in vier Schritten.

Schritt 1: TEMPUS.PRG wird aufgerufen und die Installationsdatei geladen. In der Regel dürfte die voreingestellte Zeilenlänge nicht ausreichen; aus den angebotenen Möglichkeiten wählt man dann eben *Zeilenlänge anpassen*. Um keine unangenehme Überraschung zu erleben, schaltet man gleich in den Quelltextmodus, falls dies nicht schon die Voreinstellung ist.

Schritt 2: Die Installationsdatei wird den eigenen Wünschen angepaßt. Ist dies bereits geschehen, kann dieser Schritt übersprungen werden. Zunächst speichert man die Datei unter einem neuen Namen ab (sicher ist sicher!) und beantwortet die Frage nach Übernahme des neuen Pfadnamens mit „Ja". Sämtliche Änderungen kommen so nur in die neue Datei, auch die durch eine eventuell eingestellte automatische Sicherung, und das Original bleibt unangetastet. Derart abgesichert nimmt man nun die gewünschten Änderungen vor und speichert das Ergebnis ebenfalls ab.

Schritt 3: Mit der Funktion *Quelldaten übersetzen* aus dem Menü *Parameter* werden die Angaben der im aktuellen Textfenster angezeigten Installationsdatei in das laufende Programm eingebaut und damit sofort wirksam. Sollte Tempus beim Übersetzen einen Fehler entdecken, meldet er dies und setzt den Zeiger auf die fehlerhafte Stelle. Gibt es keine Fehler (mehr), kann man auf ein anderes Textfenster umschalten und anhand einer Übungsdatei prüfen, ob auch alles so funktioniert, wie man es sich vorgestellt hat. Andernfalls wird die Installationsdatei geändert und erneut übersetzt.

Schritt 4: Zuletzt müssen noch die im laufenden Programm vorgenommenen Änderungen in die Programmdatei gespeichert werden, denn nur dann stehen sie auch beim nächsten Programmaufruf noch zur Verfügung. Dies geschieht mit Hilfe der Funktion *Param. & sichern...* des Menüs *Parameter*, in der erscheinenden Auswahlbox ist dabei der Knopf *Tastaturinstallation*, wie in Bild 3 gezeigt, anzuklicken. Wer sich hier nicht ganz sicher fühlt, dem gibt Abschnitt 11.1 des Tempus-Handbuchs ausführlich Auskunft. Dort erfährt man auch, zu welchen Installationsgruppen die anderen dem Benutzer zugänglichen Parameter gehören, so daß man deren Änderung auch gleich mit abspeichern kann.

Bei der eben beschriebenen Änderung der Tastaturbelegung kann man natürlich stets mit der gesamten Installationsdatei arbeiten. Andererseits wertet Tempus beim Übersetzen der Quelldaten stets nur die Informationen im aktuellen Textfenster aus und ändert die zugehörigen Tastenbelegungen; alle anderen Belegungen und Einstellungen werden nicht angetastet. Daher darf man die Zeilen der gewünschten Tastenkombinationen in eine separate Datei kopieren und nur noch mit dieser kleineren Datei arbeiten. Dies reduziert unerwünschte Seiteneffekte auf ein Minimum. Weil diese Arbeitsweise besonders übersichtlich ist, wollen wir auch in diesem Artikel so vorgehen.

Die Quelldaten der in Bild 1 beschriebenen Änderungen speichern wir in der Datei WS.INS ab. Bild 4 zeigt den Inhalt dieser Datei. Durch einen Stern wird bei Tempus-Quelldaten der Rest der Zeile als Kommentar markiert.

Die TeX-Version

Zur flüssigen Eingabe verwendet man hier am besten den Fließtext-Modus und bei maximaler Fenstergröße als Zeilenlänge 77 Zeichen. Die abschließenden Leerstellen oder CR-Zeichen einer Zeile bleiben so gerade noch sichtbar, und die Fensterbreite wird optimal ausgenutzt.

Zusätzlich wird über die Tastaturinstallation eine bequeme Eingabe von Standard-TeX- und LaTeX-Befehlen ermöglicht. Nach dem vorigen, wie auch nach einer Lektüre des Tempus-Handbuchs, ist jetzt klar, wie man dazu Funktionstasten oder noch freie Tastenkombinationen mit festen Zeichenfolgen belegt. Hier kann sich nun jeder nach eigenem Geschmack einrichten, am besten führt man während einiger Stunden Eingabearbeit eine kleine Liste über häufig vorkommende Textbausteine und überlegt sich anschließend eine sinnvolle Zuordnung zu den möglichen Tastenkombinationen. Für große Textbausteine, die das Entfernen der rechten Hand aus der Grundstellung rechtfertigen, bieten sich insbesondere die Tasten des rechts abgesetzten kleinen Ziffernfeldes an. Gelegentlich erscheint es sinnvoll, eine Tastenkombination zu verwenden, die von Tempus bereits belegt ist. Hier sollte man allerdings nicht unnötig viele Skrupel haben. Wie unser Beispiel WS.INS zeigt, können solche Funktionen leicht umgelegt werden, und man ist oft auch damit zufrieden, daß sie zukünftig nur mit der Maus über die Menüleiste erreichbar sind. Wir beschränken uns im folgenden darauf, Ihnen einige trickreiche Variationen, die sich in der Praxis bewährt haben und die Sie im Handbuch wohl kaum finden werden, exemplarisch vorzustellen.

T19	:=	„ {„	*[Control][8]	[{]
T20	:=	„ }“	*[Control][9]	[}]
T29	:=	16602	*[Control][E]	Zeiger nach oben
T42	:=	16600	*[Control][S]	Zeiger nach links
T43	:=	16598	*[Control][D]	Zeiger nach rechts
T50	:=	„ \“	*[Control][Ö]	[\]
T55	:=	16604	*[Control][X]	Zeiger nach unten
T145	:=	16540	*[Shift][Alternate][X]	Beenden mit Sicherheitsabfrage
T209	:=	16448	*[Alternate][E]	Suchen & Ersetzen

Bild 4: Die Datei WS.INS



Bild 5: Nach Aufruf des Makros und Eingabe der Umgebung



Bild 6: Nach Ende des Makros

\begin / \end - Makro, Teil 1

Taste(n) oder Funktionsnummer	Aktion
\begin{	Texteingabe
16600	Zeiger nach links
16470	Blockanfang setzen
16598	Zeiger nach rechts
}	Texteingabe
16472	Blockende setzen
[Return]	Texteingabe (Zeilenende)
16600	Zeiger nach links
16600	Zeiger nach links

\begin / \end - Makro, Teil 2

Taste(n) oder Funktionsnummer	Aktion
16598	Zeiger nach rechts
16598	Zeiger nach rechts
[Return]\end	Texteingabe
16394	Block kopieren
16388	Blockmarken löschen
16602	Zeiger nach oben
16602	Zeiger nach oben

Bild 7: Realisierung des „\begin“ / „\end“ - Makros

Einfache Umgebungen

Sehr häufig verwendet man in LaTeX sogenannte Umgebungen, die von Anweisungen der Form „\begin{...}“ und „\end{...}“ eingerahmt sind. Um nun nicht für jede Umgebung eine eigene Tastenkombination opfern zu müssen, die man sich dann ja auch noch alle zu merken hätte, realisieren wir in Tempus ein Makro mit Parameterübergabe. So etwas ist zwar offiziell nicht vorgesehen, aber das macht die Sache ja nur interessanter. Alles, was man dazu braucht, sind zwei Tastenkombinationen und die Blockmarkierungen. Beim Aufruf des Makros gesetzte Blockmarkierungen sind daher anschließend verschwunden. Das Makro soll mit der Tastenkombination „Control,[,]“ eingeleitet werden, danach ist als Parameter der Name der Umgebung einzutippen, die Eingabe wird mit der Kombination „Alternate,[,]“ beendet.

Das Makro fügt an der aktuellen Zeigerposition den „\begin“- und den „\end“-Befehl der Umgebung ein, schiebt eine Leerzeile dazwischen und setzt darauf den Zeiger. Man wird allerdings zukünftig auf die netten Fehlermeldungen verzichten müssen, die mittels einer fehlenden „\end“-Anweisung oder durch das Anweisungspaar „\begin{tabular}“ und „\end{tabbing}“ so mühelos gelangen. Bild 5 zeigt den Bildschirm, nachdem das Makro mit „Control,[,]“ eingeleitet und die Umgebung „verse“ eingetippt wurde. Der Name der Umgebung erscheint dabei automatisch im Blockbereich. Bild 6 zeigt den Zustand nach Abschluß des Makros mit „Alternate,[,]“. In Bild 7 sehen Sie, wie ein solcher Mechanismus mit Hilfe von Tempus-Funktionen realisiert wird.

Marken in TeX

Eine sehr mächtige Möglichkeit von LaTeX ist das automatische Führen von Querverweisen mit dem Anweisungspaar „\label“, das Marken setzt, und „\ref“, das auf diese verweist. Bei längeren Texten und vielen Marken verliert man leicht die Übersicht. Ein guter Trick ist es, für jede eingeführte Marke einen Eintrag in die Glossary-Datei zu machen. Dies ist eine von LaTeX auf Wunsch geführte Hilfsdatei, die man mit dem Befehl „\makeglossary“ anfordert. Zur Demonstration erzeugen wir hier für jede Marke in der Glossary-Datei eine Zeile einer dreispaltigen Tabelle, die in der ersten Spalte die Marke, in der zweiten Spalte eine Erläuterung dazu und in der dritten Spalte die Seitennummer enthält. Die so hergestellte Hilfsdatei kann man, nachdem die eigent-

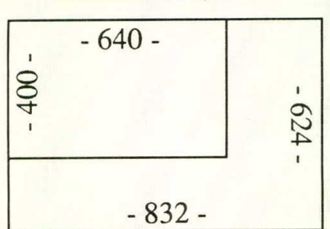
Mega Screen

Ein Monitor für alle Auflösungen

für den Mega ST erweitert niedrige, mittlere und hohe Auflösung, z.B.: 640x350 in mittlerer Auflösung, 832x624 in hoher Auflösung. Die Auflösung kann vom Desktop aus geändert werden. Volle Kompatibilität auch bei Programmen, die nur mit der Original-Auflösung arbeiten, da diese eingestellt werden kann.

Leistungen:

- Erhöhung der darstellbaren Pixelauflösung des ATARI Mega ST in Verbindung mit „Multisync" Monitoren
- Sämtliche Original-ST-Auflösungen mit 75 - 90 Hz (je nach verwendetem „Multisync" Monitor). Somit flimmerfrei auch bei Farbprogrammen
- Bis zu 24 verschiedene Auflösungen zur Auswahl (Auflösung vom Desktop aus änderbar)
- Konfigurationsprogramm zur individuellen Einstellung der verschiedenen Auflösungen
- Umschalten zwischen Farb- und Monochrombetrieb mit Auto-Monitor-Switchbox vom Desktop aus möglich
- Mehr als doppelte Anzahl an Bildpunkten auf dem Bildschirm
- Volle Nutzung des Blitters bei jeder Auflösung
- Integrierter, abschaltbarer Bildschirmschoner
- Auch Spiele laufen, da die Grafikkarte abschaltbar ist
- Läuft mit TOS 1.2 und TOS 1.4
- Getestet in C'T 3/90



Die preiswerte Grafikkarte
DM 298,-

That's PiXel

Nur ein Malprogramm?

Es gibt für den ST ca. 200 „Malprogramme". Davon sind etwa 25 in der Lage, auch Grafiken, die größer als der Bildschirm sind, zu bearbeiten. Die wenigsten erlauben einen bequemen Umgang mit dem Bild. Kein weiterer Kommentar!

- Bildgröße nur durch Speicher begrenzt
- **Bildmanipulation nicht durch Monitor begrenzt**
- Montage mehrerer Bilder, auch aus Fremdprogrammen zu einem großen Bild
- Maus- und Tastaturbedienbar
- Viele praktische Hilfs- und Zeichenfunktionen
- UNDO Puffer (abschaltbar für größte Bilder)
- Automatische Abschaltung der Ganzseitendarstellung, falls das Bild auf den Monitor paßt
- Läuft unter SM124, SM194, OverScan! und mit MegaScreen
- Import verschiedener Bildformate (IMG, PI3, PAC,
- DOO, TPI, PIC,...) die Liste der Treiber wird laufend ergänzt
- Bilder einfach und schnell bearbeiten, die ideale Ergänzung zur Scannersoftware
- Die ideale Ergänzung zu vielen Programmen am Markt

Das Grafiktool mit Konzept!

- **Gleichzeitig** und daher **übersichtlich** bearbeitet man Ganzseitendarstellung, Lupe und 1 : 1-Darstellung
- **Automatisches Verschieben** erspart das sonst mühevolle Scrollen, während man eigentlich zeichnen möchte. Die Geschwindigkeit ist per Tastendruck auf Alternate, Shift und Control wählbar
- **Kürzeste Ladezeit des Programms!** Erspart lästiges Warten beim Programmwechsel
- **Schnelles Scrollen,** Bild laden, sichern, Block kopieren, verschieben,... ermöglicht fließendes Arbeiten. Unkomplizierte, **einfache Bedienung** mit Übersicht
- **Kurze Einarbeitung** wird durch das verständliche Konzept ermöglicht. Ausdruck auf allen gängigen Druckern in verschiedenen Auflösungen möglich. Auf Wunsch Page-Preview vor Druck

– Alle genannten Preise unverbindliche Preisempfehlungen –

DM 148,-

Heim Verlag
Heidelberger Landstraße 194
6100 Darmstadt-Eberstadt
Telefon 0 61 51 - 5 60 57

Hotline
Info + Betreuung
COMPO SOFTWARE GMBH
Ritzstr. 13
5540 Prüm
Tel.: 0 65 51 / 62 66

BESTELL-COUPON

Bitte senden Sie mir: _____ MegaScreen à DM 298,-
_____ That's Pixel à DM 148,-
zzgl. Versandkosten DM 6,- (Ausland DM 10,-) unabhängig von der bestellten Stückzahl

Name, Vorname ______________________
Straße, Hausnr. ______________________
PLZ, Ort ______________________
Oder benutzen Sie die in ST-Computer eingeheftete Bestellkarte

In Österreich:
RRR EDV GmbH
Dr. Stumpfstraße 118
A-6020 Innsbruck

In der Schweiz:
Data Trade AG
Landstraße 1
CH - 5415 Rieden-Baden

liche Datei mit TeX gesetzt wurde, in einer weiteren Datei mit „\input“ innerhalb einer geeigneten „tabular“-Umgebung einlesen und sich so eine übersichtliche Liste aller im Text verwendeten Marken ausgeben lassen. Bild 8 zeigt eine solche Rahmendatei, die eine beim Setzen von BEISPIEL.TEX entstandene Hilfsdatei BEISPIEL.GLO einliest und zu einer Tabelle verarbeitet. Die unterstützende Tempus-Funktion soll wieder als Makro mit Parameterübergabe organisiert werden.

Mit der Tastenkombination „Alternate,[G]“ wird das Makro aufgerufen, danach die Marke eingetippt. Bild 9 zeigt den Bildschirm nach diesem Vorgang. Bis hierher wird nur mit „\label“ eine Marke gesetzt; wenn man damit zufrieden ist, braucht man nichts weiter zu tun. Um nun allerdings einen Eintrag in die Glossary-Datei zu erzeugen, wird noch die Tastenkombination „Shift,Alternate,[G]“ gedrückt, die den entsprechenden Eintrag erzeugt. Der Zeiger steht anschließend an der Stelle, an der ein Kommentar zur Marke eigegeben wird, wie in Bild 10 zu erkennen ist. In Bild 11 geben wir die Konstruktion des Makros mit Tempus-Befehlen an. Bild 12 enthält die für die beiden obigen Makros erforderlichen Quelldaten, in der Datei TEX.INS zusammengefaßt. Die Darstellung erfolgt diesmal als Bildschirmfoto, damit Sie die einzugebenden CR-Zeichen besser erkennen können. Sie sehen, wie bei Zeichenketten jedes Zeichen einzeln und mit vorangestelltem Leerzeichen eingegeben werden muß.

Die FORTRAN-Version

Durch die unter FORTRAN77 vorgeschriebene starre Aufteilung der Zeilen in Felder mit besonderer Bedeutung ergeben sich an einen Editor grundsätzlich andere Anforderungen als bei anderen Programmiersprachen oder gar Fließtext. Die enormen Variationsmöglichkeiten von Tempus gestatten es jedoch, auch für diesen Fall einen optimal angepaßten Editor einzustellen. Die vorgenommenen Modifikationen für TeX und FORTRAN schließen sich allerdings teilweise gegenseitig aus, so daß wir hier besser eine eigene Programmdatei herstellen. Zu diesem Zweck wird TEMPUS.PRG etwa in die Datei F-TEMPUS.PRG (welchen FORTRAN-Compiler verwendet der Autor wohl?) umkopiert. Mit einer derart umbenannten Programmversion kann man ohne Schwierigkeiten arbeiten, selbst die Funktion *Param. & sichern...* funktioniert fehlerfrei.

Ausgehend von der Tempus-Grundversion werden nun einige Änderungen ein-

```
%  Glo.TeX  (23 Jul 90)  Volker Kurz
%
\documentstyle{article}
%
%  Je nach LaTeX-Version wird für Einträge in der glossary-Datei
%  entweder  \indexentry  oder \glossaryentry  verwendet.
%  Zur Sicherheit definieren wir beides:
%
\def\indexentry#1#2{#1 & #2\\}
\def\glossaryentry#1#2{#1 & #2\\}
%
\begin{document}
%
\noindent\begin{tabular}{clr}
\hline
Marke & Bedeutung & Seite \\\hline
\input{Beispiel.glo}
\hline
\end{tabular}
\end{document}
```

Bild 8: Rahmen-Datei GLO.TEX

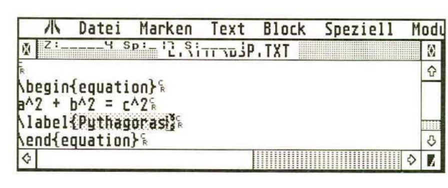


Bild 9: Nach Auruf des Makros und Eingabe der Marke

```
 Datei  Marken  Text  Block  Speziell  Modu
 Z:____4  Sp:_42  S:      P.TXT

\begin{equation}
a^2 + b^2 = c^2
\label{Pythagoras}\glossary{{Pythagoras}& }
\end{equation}
```

Bild 10: Nach Ende des „\label“-Makros

```
\label - Makro, Teil 1

Taste(n) oder Funktionsnummer    Aktion
_____________________________
\label{                          Texteingabe
16600                            Zeiger nach links
16470                            Blockanfang setzen
16598                            Zeiger nach rechts
}                                Texteingabe
16472                            Blockende setzen
16600                            Zeiger nach links

\label - Makro, Teil 2

Taste(n) oder Funktionsnummer    Aktion
_____________________________
16598                            Zeiger nach rechts
\                                Texteingabe
16600                            Zeiger nach links
16472                            Blockende setzen
16598                            Zeiger nach rechts
glossary{                        Texteingabe
16394                            Block kopieren
&[Leerstelle]}                   Texteingabe
16600                            Zeiger nach links
```

Bild 11: Realisierung des „\label“-Makros

```
  Datei  Marken  Text  Block  Speziell  Modus  Parameter        E  12:39:02
 Z:____12  Sp:__0  S:____         E:\TTF\TEX.INS
T61  :=  " \" " b" " e" " g" " i" " n" " {" 16600 16470 16598 " }" 16472
     " CR" 16600 16600
     *[Control][,]    \begin{.}
T135 := 16598 " \" 16600 16472 16598 " g" " l" " o" " s" " s" " a" " r" " y"
     " {" 16394 " &" "  " " }" 16600
     *[Shift][Alternate][G] \label-Zeile, 2. Teil
T225 :=  " \" " l" " a" " b" " e" " l" " {" 16600 16470 16598 " }" 16472
     16600
     *[Alternate][G]  \label-Zeile, 1.Teil
T241 :=  16598 16598 " CR" " \" " e" " n" " d" 16394 16388 16602 16602
     *[Alternate][,]  \end{.}
```

Bild 12: Quelldaten in TEX.INS

gebracht. Zunächst einmal ist der automatische Zeilenumbruch in FORTRAN nicht erwünscht, so daß besser auf Quelltext umgeschaltet wird. Weiter ist es ganz praktisch, die Zeilenlänge auf die in FORTRAN maximal erlaubten 72 Zeichen einzustellen. Ängstliches Abzählen hat damit endlich ein Ende, ebenso die allseits beliebten Syntaxfehler aufgrund überlanger und vom Compiler gewaltsam verkürzter Anweisungszeilen. Automatisches Einrücken ist ebenfalls von Vorteil und wird daher eingeschaltet. Dazu gehört natürlich auch eine alternative Form der Zeilenfortschaltung, die die aktuelle Einrückung gerade nicht übernimmt, doch davon später mehr. Diese Parameter werden nun für alle vier Fenster eingestellt, wegen der Breite von 72 Zeichen kann man die Fenster sogar ein wenig verkleinern und seitlich versetzt anordnen. Auch hier wollen wir die einfache Zuordnung fester Zeichenketten wieder dem Leser überlassen und Ihnen dafür zwei besondere Leckerbissen vorstellen.

DO-Schleifen

Schleifen programmiert man in FORTRAN meist mit der DO-Anweisung, wobei zur besseren Übersicht die Schleife mit einer CONTINUE-Anweisung enden und der Schleifenkörper eingerückt werden soll. Die CONTINUE-Anweisung trägt die Anweisungsnummer, die in der DO-Anweisung angegeben ist. Aufgrund dieser Fülle formaler Anforderungen eignet sich die DO-Schleife vorzüglich zur automatisierten Eingabe.

Die Implementierung geschieht natürlich wieder in Form eines Makros, diesmal verwenden wir die Tasten des rechts abgesetzten Ziffernblocks. Die einleitende Tastenkombination „Shift,[Ziffernblock (]“ schreibt das DO, eine Leerzeile mit um drei Stellen verlängerter Einrückung, die CONTINUE-Zeile, die genauso weit eingerückt ist wie die DO-Zeile, und ermöglicht die Eingabe der Anweisungsnummer innerhalb eines markierten Blocks. Den Abschluß des Makros ruft man mit „Shift,[Ziffernblock)]“ auf. Dabei werden in die CONTINUE-Zeile die Anweisungsnummer eingefügt und der Zeiger in die DO-Zeile wieder hinter der Anweisungsnummer plaziert. Den Zustand nach Aufruf des Makros und Eingabe der Anweisungsnummer sehen Sie in Bild 13, das fertige Ergebnis in Bild 14. Um die verschiedenen Einrückungen zu veranschaulichen, wurde ein anderer Zeichensatz verwendet, bei dem die Leerstellen durch Punkte dargestellt sind.

Eine kleine Einschränkung gibt es bei unserem Makro allerdings: Die DO-Zeile selbst darf keine Anweisungsnummer tragen, gegebenenfalls ist diese anschließend einzufügen oder ein CONTINUE voranzustellen. In Bild 15 ist die Umsetzung des Makros in Tempus-Funktionen erläutert; besonders interessant ist im zweiten Teil die Einfügung verschieden langer Anweisungsnummern bei gleichzeitiger Erhaltung der Einrücktiefe in der CONTINUE-Zeile.

Auf gleiche Weise lassen sich auch Block-IF-Strukturen behandeln; da hierbei aber keine Anweisungsnummern vorkommen, ist alles ein wenig einfacher und bleibt daher unseren Leserinnen und Lesern als leichte Übung überlassen.

```
 Datei  Marken  Text  Block  Spezi
......N=10
......DO 111
.........
......CONTINUE
```

Bild 13: DO-Makro nach Eingabe der Anweisungsnummer.

```
 Datei  Marken  Text  Block  Spezi
......N=10
......DO 111 
.........
111...CONTINUE
```

Bild 14: DO-Makro nach getaner Arbeit

DO - Makro, Teil 1

Taste(n) oder Funktionsnummer	Aktion
16472	Blockende setzen
16566	Zeiger an Zeilenanfang
16470	Blockanfang setzen
16568	Zeiger an Zeilenende
[Return]	Texteingabe
16394	Block kopieren
3*[Leerstelle],[Return]	Texteingabe
16394	Block kopieren
CONTINUE	Texteingabe
2*16602	2 * Zeiger nach oben
DO,4*[Leerstelle]	Texteingabe
5*16600	5 * Zeiger nach links
16470	Blockanfang setzen
16598	Zeiger nach rechts

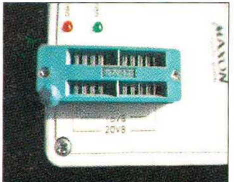
DO - Makro, Teil 2

Taste(n) oder Funktionsnummer	Aktion
16570	Zeiger zum Wortanfang links
16470	Blockanfang setzen
5*16598	5 * Zeiger nach rechts
16472	Blockende setzen
2*16604	2 * Zeiger nach unten
16566	Zeiger an Zeilenanfang
5*16586	5 * Zeichen unter Zeiger löschen
16394	Block kopieren
2*16602	2 * Zeiger nach oben
16568	Zeiger an Zeilenende
3*16578	3 * Zeichen links löschen
16388	Blockmarken löschen

Bild 15: Realisierung des DO-Makros

Einrücken und Ausrücken

Da wir uns in F-TEMPUS für automatisches Einrücken entschieden haben, ist das Einfügen von Kommentarzeilen bei der laufenden Eingabe etwas unhandlich. Wir lösen dieses Problem, indem wir der bisher noch unbenutzten Tastenkombina-



GAL-Programmiergerät MGP 16/20

Entwicklungssystem für Logikschaltungen

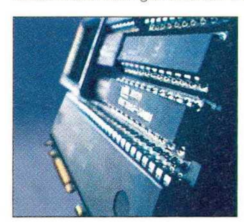
Leistungsstarkes Programmiergerät für die Realisierung logischer Schaltungen (NOR-, NAND-, NOT-,... Gatter) mit den gängigen GAL-Typen 16v8 und 20v8 und deren A-Typen. Das Gerät wird an die Druckerschnittstelle (parallel - Centronics) angeschlossen. Die menügesteuerte Software ermöglicht ein bequemes und sicheres Arbeiten. Integrierter 2-Pass-Logic-Compiler, der logische Gleichungen in JEDEC-Dateien übersetzt. Optimierung der Gleichung nach Quine-McCluskey.

Bestellnr.: 890900 Fertiggerät .... DM 229,-*
Bestellnr.: 890901 Platine, Software .... DM 129,-*

Mach 16

16 MHz für alle ST

Sollte Ihnen Ihr ST zu langsam sein, können Sie Ihn mit dieser Beschleunigerkarte auf Trab bringen. Prozessor 68000 mit 16 MHz Taktrate, 16 kByte O-Waitstate-Cache-Memory, Sockel für optionalen Arithmetik-Coprozessor MC68881, Einbau in 260ST, 520ST, 520ST+, 1040ST sowie alle Mega STs möglich, Beschleunigung im Praxisbetrieb um Faktor 1,85.



Bestellnr.: 900820 Fertiggerät .... DM 695,-*

HINTERM HORIZONT GEHT'S WEITER.

Junior Prommer

EPROM-Programmiergerät



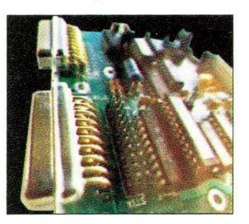
Programmiert alle gängigen EPROM-Typen und deren CMOS-Typen (2716-27011). Komfortable Software mit Zerlegung in High- und Low-Byte, 5 Programmieralgorithmen, Hex-/ASCII-Monitor mit vielen Edierfunktionen. Leichtes Erstellen von EPROM-Bänken durch Software-Unterstützung. Mit optionalem Adaptersockel Mega-Modul lassen sich auch 32pol. EPROMs (27010-27080) brennen.
Das Gerät wird an die Druckerschnittstelle (parallel - Centronics) angeschlossen.

Bestellnr.: 880310 Fertiggerät .... DM 229,-*
Bestellnr.: 880311 Platine, Software .... DM 59,-*
Bestellnr.: 880312 Leergehäuse .... DM 39,90*
Bestellnr.: 880313 Zusatzadapter Mega Modul DM 99,-*

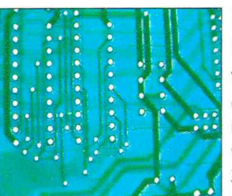
MSA

SCSI-Adapter

Schneller Adapter zum Anschluß von SCSI-Geräten an den Atari ST. Übertragungsraten bis zu 1000 kByte/sec., macht das angeschlossene SCSI-Gerät uneingeschränkt bootfähig, kompatibel zu allen erhältlichen SCSI-Festplatten, unterstützt alle SCSI-Kommandogruppen und hat einen gepufferten DMA-Bus, Anschluß von max. 4 SCSI-Geräten, Hardware-Schreibschutz, inkl. Festplattentreiber.



Bestellnr.: 900810 Fertiggerät .... DM 259,-*
Bestellnr.: 900811 Platine, GALs,
Software, Platine .... DM 149,-*



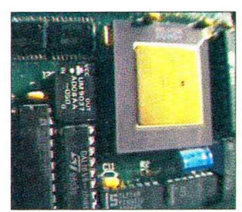
DPE

Drucker-Port-Expander

Vielseitige Erweiterungsschaltung für den Drucker-Port des ATARI ST. Mit ihm läßt sich elektronisch zwischen 3 verschiedenen Geräten umschalten. Zwei zusätzliche Joystick-Buchsen zur Stromversorgung externer Geräte /z.B. Junior Prommer oder MGP). Durch eigene Treiberbausteine können auch Geräte in größerer Entfernung angeschlossen werden.

Bestellnr.: 900800 Platine und Bauanleitung ...... DM 49,-*

Fordern Sie ungeniert unseren Prospekt an!



Board 20

68020-Accelerator

Mit dieser Beschleunigungskarte wird Ihr ST zur 32-Bit-Workstation. Prozessor MC68020 mit 16 MHz Taktrate, 32 kByte Cache-Memory mit 32 Bit Busbreite, optimierte Cache-Verwaltung, höchste Kompatibilität durch optimiertes TOS 1.6 im ROM, Beschleunigung im Praxisbetrieb um ca. 360%.

Bestellnr.: 900830 Fertiggerät .... DM 1895,-*

*Alle Preise sind unverbindliche Preisempfehlungen
Versandkosten Inland: DM 7,50
Versandkosten Ausland DM 10,-
Auslandsbestellungen nur gegen Vorauskasse



MAXON Computer GmbH • Schwalbacher Straße 52 • W-6236 Eschborn
Tel. 06196/481811 • Fax 06196/41885




DISKETTEN-SERVICE
Absender bitte deutlich schreiben
Vorname / Name
Straße / Nr.
PLZ / Ort
Telefon
Postkarte
Bitte freimachen
MAXON Computer
Schwalbacher Straße 52
D-6236 Eschborn
ST COMPUTER
Kontaktkarte
ST COMPUTER
Kontaktkarte
Bitte Adresse der Firma, bei der Sie Informationen, oder etwas bestellen möchten, auf der rechten Seite eintragen → → → → → → → → → →
Absender
(Bitte deutlich schreiben)
Vorname/Name
Beruf
Straße/Nr.
PLZ/Ort
Telefon Vorwahl/Rufnummer
Postkarte
Bitte freimachen
Firma
Straße/Postfach
PLZ
Ort
ST COMPUTER
ABO
ST COMPUTER
ABO
Absender = Rechnungsadresse
(Bitte deutlich schreiben)
Vorname/Name
Straße/Nr.
PLZ/Ort
Postkarte
Bitte freimachen
Heim Verlag
Heidelberger Landstr. 194
6100 Darmstadt 13

ST COMPUTER
ATARI

Public Domain Service

Bitte senden Sie mir folgende PD-Disketten / Sonderdisks

Datum / Unterschrift

Zahlung erfolgt:

___ Vorauskasse (per Scheck) ___ per Nachnahme

- Pro PD-Diskette entfällt ein Unkostenbeitrag von DM 10.-
- Preis der Sonderdisks je nach Angabe
- Bei Nachnahme zuzüglich DM 4.- Nachnahmegebühr
- Versandkosten (Porto und Verpackung) DM 5.- (Ausland DM 10.-)
- Auslandsbestellungen nur gegen Vorauskasse möglich

Ab 5 Disketten versandkostenfrei

SONDERDISK

Bitte beachten Sie, daß Sonderdisks ein Copyright besitzen und nicht weiterkopiert werden dürfen.

DISKETTEN-SERVICE

ST COMPUTER

☐ Ich bitte um weitere Informationen

☐ Ich gebe folgende Bestellung auf

in Bezug auf Ihre Anzeige in ST-Computer Heft ______ Seite ___

Menge	Produkt/Bestellnummer	DM	gesamt DM

Datum, Unterschrift (für Jugendliche unter 18 Jahren der Erziehungsberechtigte)

Abgesandt am:

ST COMPUTER

Firma:

Bemerkungen:

ST COMPUTER

Abonnement

Kd.-Nr.:

Ja, bitte senden Sie mir die ATARI-Computer Fachzeitschrift ab der nächsten erreichbaren Ausgabe für mindestens 1 Jahr (11 Hefte) zum ermäßigten Preis von jährlich DM 80,- frei Haus.
(Ausland: nur gegen Scheck-Voreinsendung DM 100,- Normalpost, DM 130,- Luftpost)
Der Bezugszeitraum verlängert sich nur dann um ein Jahr, wenn nicht 6 Wochen vor Ablauf des Abonnements gekündigt wird.

Geschenk- bzw. Lieferadresse:

Vorname

Name

Straße / Postfach

Land PLZ Ort

Datum Unterschrift

Gewünschte Zahlungsweise bitte ankreuzen

☐ Bequem und bargeldlos durch Bankeinzug

Konto-Nr. BLZ

Institut Ort

☐ Ein Verrechnungsscheck über DM ____ liegt bei.

☐ Vorauskasse per Zahlung auf unser Post-scheck-Konto Ffm, BLZ 500 100 60, Kto.-Nr. 5537-602

Diese Vereinbarung kann ich innerhalb von 8 Tagen beim Heim-Verlag, Heidelberger Landstr. 194, 6100 Darmstadt-Eberstadt widerrufen. Zur Wahrung der Frist genügt die rechtzeitige Absendung des Widerrufs. Ich bestätige die Kenntnisnahme des Widerrufsrechts durch meine 2. Unterschrift.

Datum 2. Unterschrift

ST COMPUTER

ABO

Diese Vereinbarung kann ich innerhalb von 8 Tagen beim Heim-Verlag, Heidelberger Landstr. 194, 6100 Darmstadt-Eberstadt widerrufen. Zur Wahrung der Frist genügt die rechtzeitige Absendung des Widerrufs. Ich bestätige die Kenntnisnahme des Widerrufsrechts durch meine 2. Unterschrift.

tion „Control,[Return]“ die beiden Funktionen *CR-Zeichen einfügen* und *Zeilenanfang bis Zeiger löschen* zuordnen. Der Zeiger steht danach in der ersten Spalte einer neuen Zeile, und es gibt keine überflüssigen Leerzeichen. Ebenfalls nützlich ist eine Funktion zum Ausrücken, die also die Einrückung in der nächsten Zeile um eine Stufe, das sind hier drei Zeichen, reduziert. Dazu verwenden wir das bisher ebenfalls ungenutzte „Alternate,[Return]“, das wir mit der Funktion *CR-Zeichen einfügen* und anschließendem dreimaligem *Zeichen links vom Zeiger löschen* belegen. Die Datei FORTRAN.INS mit den Quelldaten zur Tastenbelegung zeigen wir Ihnen in Bild 16 wieder als Bildschirmfoto, damit die CR-Zeichen besser zu erkennen sind.

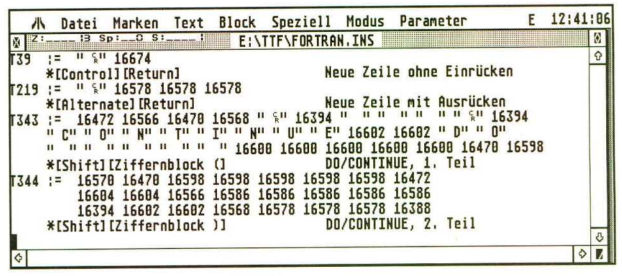


Bild 16: Quelldaten in FORTRAN.INS

Variationen

Mit WS.INS, TEX.INS und FORTRAN.-INS haben wir Ihnen nun drei verschiedene Möglichkeiten der Tastaturanpassung vorgestellt. Natürlich können Sie WS.INS mit jeder der beiden anderen Möglichkeiten kombinieren, dazu werden entweder die Tempus-Quelldateien nacheinander übersetzt oder in einer gemeinsamen Quelldatei zusammengefaßt. Sollten Sie die Anpassungen selbst noch erweitern, dann legen Sie sich am besten von Zeit zu Zeit eine Sicherheitskopie der aktuellen Quelldatei an, um so bei Bedarf mit Hilfe der Originaldatei TEMPUS.PRG jederzeit zu einer beliebigen Zwischenstufe zurückkehren zu können.

Wer mehr will...

Insbesondere für das C-TeX von Stefan Lindner wären noch weitergehende Modifikationen denkbar: Man könnte etwa den von TEMPUS verwendeten Zeichensatz durch Aufnahme besonderer TeX-Symbole ändern und diese mit Hilfe der vom Lindnerschen TEX.TTP gelesenen Parameterdatei TEXSETUP dann in TeX-Befehle umsetzen. Hierdurch läßt sich schon fast eine Art TeX-Preview erreichen. Der Preis ist aber sehr hoch, denn während sonst TeX-Quelltexte grundsätzlich über alle Maschinen kompatibel sind, braucht man jetzt nur eine andere TeX-Implementierung auf dem gleichen Rechner einzusetzen, um sich eine Lektion in Sachen Portabilität erteilen zu lassen. Natürlich könnte TEMPUS auch hier wieder helfen, nämlich über die Druckerinstallation und die Funktion *Drucken in Datei...*, aber das würde an dieser Stelle wirklich etwas zu weit führen.

Wir haben uns also auf solche Änderungen beschränkt, die die Eingabe umfangreicher Befehle zwar erleichtern, nicht aber den Quelltext durch zusätzliche Symbole zwar übersichtlicher, dafür aber auch inkompatibel machen. Oberste Maxime ist, daß die erstellten Texte von jedem beliebigen anderen Editor weiterverarbeitet werden können.

ASCII und Fließtext

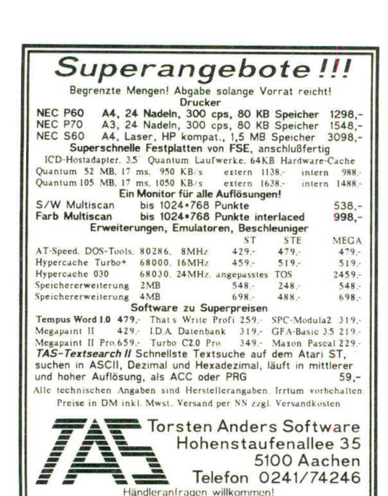
Die einzige Abweichung von dieser strengen Regel gestatten wir uns durch die Eingabe im Fließtextmodus. Hier werden von TEMPUS am Absatzende und am Ende von Zeilen, die nicht umbrochen werden sollen, zusätzliche CR-Zeichen eingefügt. Diese kann man jedoch im gesamten Text in Sekundenbruchteilen mit der Funktion *Zeichenredundanz* im Menü *Text* entfernen.

Andererseits läßt sich ein beliebiger ASCII-Text auch leicht gegen das Zusammenschieben im Fließtextmodus von Tempus schützen. Die Funktion *Text anpassen* im Menü *Text* fügt ein solches Zeichen automatisch am Ende all der Absätze ein, die vom nächsten Absatz durch eine Leerzeile getrennt sind. Oft sind jedoch nicht alle Zeilen, die man separat halten möchte, durch Leerzeilen getrennt. In einem solchen Fall markiert man am besten jedes Zeilenende mit einem CR-Zeichen und entfernt die zusätzlich eingefügten Zeichen anschließend nur da wieder, wo sie wirklich stören. Die ursprüngliche Zeilenaufteilung eines Textes bleibt somit überall dort erhalten, wo man sie nicht explizit widerruft.

Das Einfügen eines zusätzlichen CR-Zeichens an jedem Zeilenende erreicht man wie folgt: Da die Parameter, mit denen Tempus Texte speichert, im folgenden kurzfristig verändert werden, sollte man Tempus entweder neu starten oder die automatische Dateisicherung vorübergehend abschalten. Anschließend lädt man den gewünschten Text und schaltet in den Fließtextmodus. Dann wird die Funktion *TEMPUS-Parameter* im Menü *Parameter* gewählt. Mit der Maus klicken Sie das Eingabefeld hinter „Zeilenendekennzeichnung“ an. Dort stehen bereits zwei Zeichen: CR und die Glocke, das Symbol für „line feed“ (ASCII 10); dahinter sieht man jetzt den Zeiger in Form eines dünnen senkrechten Strichs. Der Zeiger wird mit der Taste „Pfeil nach links“ zwischen die beiden Sonderzeichen gesetzt. Hier fügt man ein weiteres CR-Zeichen durch die Tastenkombination „Control,[M]“ ein. Anklicken des OK-Feldes beendet die Eingabe. Danach wird der Text gespeichert. Jetzt verläßt man das Programm Tempus, oder man löscht das eingefügte CR-Zeichen im Parameter-Formular sofort wieder und holt die eben abgespeicherte Datei durch *Text überladen* mit der veränderten Form in den Speicher zurück.

Mit diesem Artikel haben wir Ihnen hoffentlich einen Eindruck davon vermittelt, zu welch mächtigem Hilfsmittel Sie diesen schnellen Editor ausbauen können. Für Ihre eigenen Experimente wünschen wir Ihnen viel Spaß!

Dr. Volker Kurz



Schulmeister ST

Atari ST (Mega ST) 500 K Ram, sw - Monitor. Die Noten- und Klassenverwaltung mit Pfiff. Ein flexibles, bewährtes Konzept für Lämpels aller Schulstufen. Lassen Sie Ihren Rechner die tägliche Routinearbeit erledigen damit Sie sich Ihren pädagogischen Aufgaben widmen können. Auch für die Schweiz geeignet !



Ausführliche Information mit Freiumschlag anfordern bei:

M.Heber-Knobloch
Auf der Stelle 27
7032 Sindelfingen

> ATARI ST <

Anwendersoftware		Spielesoftware	
CCD		Star Trash	50,-
ST Pascal + V2.0x	220,-	Gunship	79,-
Tempus Editor 2.1x	100,-	Indiana Jones (Adv.)	69.-
Tempus Word	a.A.	Zak MacKraken	69,-
Assembler Tutorial	90,-	Populous	69,-
GFA		Rick Dangerous	69,-
GFA Basic 3.5 (I.+C.)	240,-	RVF Honda	69,-
GFA Assembler	140,-	Sleeping Gods Lie	69,-
Omikron		Spherical	53,-
Omikron Basic Comp.	170,-		
Mortimer, Utility	75,-	Supercharger 1MB	720,-
Appl. Syst.		Coprozessor f. Sc.	285,-
Signum!2	418,-	Turbo C (PC/TOS)	220,-
Scarabus	90,-	Turbo Pascal (PC)	350,-
Signum Revers Acc.	90,-	Traktrix	80,-
Protos	64,-	Approximationsprg. für sämtl. Fkttypen	
Fontdisketten verfügbar			

Porto: Vorkasse 4,- Nachnahme 7,- DM

Computerversand G. Thobe
Pf. 1303 - W-4570 Quakenbrück
Tel.: (05431) 5251

Einkommen-/Lohnsteuer 1990

Direkt vom Steuerfachmann. Berechnet alles. Komfortable Eingaben, jederzeit korrigierbar, aussagekräftige Ausgabe mit Hinweisen auf Steuervergünstigungen, Datenabspeicherung, Alternative Berechnungen, Berlinpräferenz, § 10e! 54-seitige ausführ. Broschüre. **Ausdruck in die Steuererklärung.** (Mantel, N, V, KSO)

Vorgestellt als Entdeckung des Monats in PC Praxis 1/91

Für Atari ST mono nur 99 DM

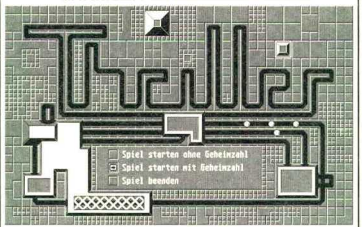
Demo-Disk 10 DM · Info gg. Porto bei

Dipl. Finanzwirt Uwe Olufs

Bachstr. 70k · 5216 Niederkassel 2

Tel.: 02208/4815 FAX/BTX 022084815

Jetzt mitmachen beim großen THRILLER Preisausschreiben!



Lösen Sie auf Ihrem ATARI ST (mind. 1 MB, S/W) die 100 gefahrvollen und kniffligen Bilder (Jo-Jo-Bilder, Scheinwerferbilder, Symmetriebilder ...)! Es lohnt sich: Der aus allen richtigen Einsendungen ermittelte Gewinner kann wählen zwischen einer Festplatte oder Bargeld im Wert von 1200,– DM. Bestellungen bitte an:

SOFT THRILL J. Verwohlt, Neuer Graben 172, 4600 Dortmund oder werktags ab 18.00 Uhr: 0251/232295

THRILLER mit Preisausschreiben 59,90 DM inklusive
THRILLER ohne Preisausschreiben 49,90 DM Versand

Der SteuerStar '90

Lohn- u. Einkommensteuer 90

50,- DM/Update 30 DM

für alle ATARI-ST sw/col

Test: ST-Magazin 2/89 :

"Der Steuerstar... nimmt ohne Zweifel einen sicheren Platz in der Reihe der Spitzensoftware für den ST ein."

Dipl. Finanzwirt J. Höfer
Grunewald 2a
5272 Wipperfürth
Tel. 02192/3368





MIDIBOX™ Multitimbral Expander: 8 Instrumente + drums

16-stimmig, 99 Sounds, 11 DrumSamples: D/A: 16bit S/N: 80dB: Das 8-Mann-Orchester: 548.- DM

Band In A Box

Generiert nach Akkord-Eingabe [C-F-G7-Am]: Piano + Bass + Drums und spielt in 24 Stilarten nach Wunsch, von Rock bis Reggae (+Midifiles): 198.-

1stTRACK Professional MIDI SEQUENCER

24 Spuren, Timing Korrektur, Event-Editor, Step-Input Copy-& Processing-Tools, Sysex/Midifiles: 158.-

Wir haben über 50 (!) Midi-Programme entwickelt. Fragen Sie uns, bevor Sie ein- oder aufsteigen. Wir informieren auch über die MIDIMUSIC Collection.



FÜR SIGNUM · SCRIPT UND TEMPUS WORD

BODONI·CASLON·AKZIDENS TRANSITIONAL·SANSSERIF & SATZ-PAKETE FÜR SPRACH-WISSEN-SCHAFTLER

RICHILDENSTR.24 · 8 MÜNCHEN 19 · TEL 089/17 45 87

SCHRIFTENKATALOG DM 5,- IN BRIEFMARKEN

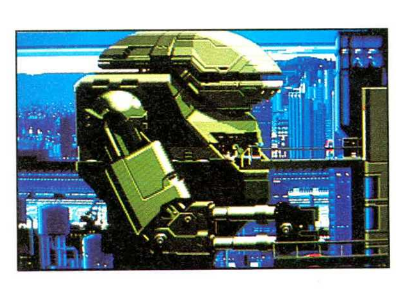
Hallo Spiele-Fans,

Anläßlich des einjährigen Firmenjubiläums veröffentlicht „Thalion“ eine Compilation mit dem Kampfsportspektakel „Chambers of Shaolin“, den beiden Ballerspielen „Warp“ und „Leavin' Terramis“ sowie dem Jump & Run Game „Seven Gates of Jambala“. Als Bonus Bonbon liegt der Verpackung ein schwarzes T-Shirt bei!



Kampf der Roboter. Zwei riesige Stahlkolosse stehen sich gegenüber, schlagen gegenseitig mit ihren Fäusten aufeinander ein und verschießen Bomben und Laserstrahlen solange, bis einer vom beiden ausgebrannt am Boden liegt. Der Gewinner bekommt zur Belohnung Geld, das er in der nächsten Runde in neue Extrawaffen investiert. Besonders zu zweit ist der Roboterkampf eine ganz spaßige Angelegenheit.

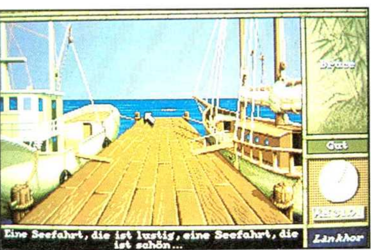




Nochmal „Thalion“! Atari ST-Soundvirtuose Jochen Hippel war im Tonstudio. Resultat: eine CD, randgefüllt mit tollen Musiken. Unter anderem gibt es „Wings of Death“ und „Dragon Flight“ in Spitzenqualität zu hören. Wem Hippel am Computer nicht reicht, der kann die CD für knapp 20 Mark bei Thalion Software in Gütersloh bestellen.



In Coktel Visions neuem Erotik-Adventure „Geisha“ verschleppt ein japanischer Mafiaboss die schöne Eve aus ihrer Pariser Yuppy-Wohnung nach Tokyo, um sie dort als Prototyp für eine neue Generation williger Sex-Roboter zu benutzen. Neben der adventureartigen Befreiungsaktion wird der Spieler mit schwachen Action-Szenen genervt: Mal muß er mit einer nackten Taucherin Perlen fischen oder mit einem Raumschiff über Kusslippen fliegen. Langweilig, einfallslos, technisch minderwertig. Von Erotik keine Spur!



So schön können Adventures sein: Grafiken wie aus dem Bilderbuch, wohlklingende Musik und Sprachausgabe. Aber auch spielerisch ist Langhors Kriminalabenteuer eine Meisterleistung: Als Dedektiv Jerome Lange wird man auf eine Karibikinsel verschlagen und soll dort das geheimnisvolle Verschwinden der kleinen Marie aufklären. Der Spieler wählt zwischen 16 Handlungsmöglichkeiten, beschattet Verdächtige und untersucht Indizien.

„Turrican II“ ist da! Wieder wird geballert, bis die Joystick-Kontakte schmoren. Es gibt mal wieder viele Extrawaffen und unheimlich große Landschaften, in denen es nur so von Widersachern wimmelt. Neu: In drei Spielabschnitten läuft man nicht mit dem Elitekämpfer Turrican durch die Gegend, sondern fliegt mit einem Raumschiff umher. Action pur! Nur für Ballerspiel-Profis geeignet!



The Secret of Monkey Island



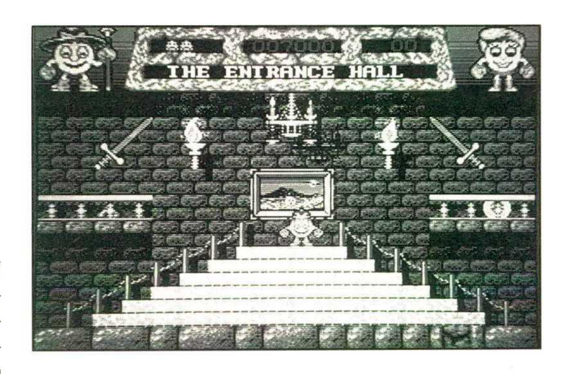
Guybrush Threepwood, der Grünschnabel mit dem zungenbrecherischen Namen, steht ganz am Anfang einer halsbrecherischen Piratenkarriere. Und wir als Spieler sind dabei beim neuesten Adventure-Spaß aus dem Hause Lucasfilm Games mit stimmungsvollen Grafiken, irre witzigen Texten, hirnerweichenden Rätseln und Calypso-Klängen vom Soundmagier Chris Hülsbeck. Im Vergleich zu anderen Lucasfilm-Spielen hat „Monkey Island" in Sachen Witz und Grafikqualität zugelegt. Gesteuert wird das komplett eingedeutschte Abenteuer per Maus und mit der für Lucasfilm Games typischen „Benutzeroberfläche", bei der man das gewünschte Verb einfach in einer Liste und das Objekt im Grafikfenster anklickt. Neu ist, daß unter den möglichen Aktivitäten bereits die wahrscheinlichste angewählt wird. Steht der Held beispielsweise vor einer verschlossenen Tür, so leuchtet in der Verbenliste automatisch „öffnen" auf. Der Spieler kann nun mit der Maus bestätigen oder einen anderen Befehl auswählen. Das Inselspiel ist in jeder Hinsicht zuvorkommend: Es hat keinerlei böse Folgen, wenn man beispielsweise das Fleisch aus der Piratenküche verliert. Geht man wieder in die Kombüse zurück, liegt dort bestimmt ein neues Stück bereit. Der Spielfigur selber kann auch nicht viel passieren. Sie ist nahezu unsterblich, es sei denn, sie stellt sich wirklich zu dumm an und stürzt sich in gefährlichen Situationen in den Selbstmord. Daraus darf der Leser jedoch nicht schließen, daß er das Geheimnis der Affeninsel blitzschnell lösen könnte. Wir lernen Guybrush am Strand der Karibikinsel Melee kennen. Er ist soeben eingetroffen. Sein Kopf ist voll von Illusionen über sein zukünftiges Leben als Pirat. Dabei weiß er noch gar nicht, was auf ihn zukommt. Um herauszufinden, wie man in diesem Job anfängt, sollte er sich von Fachleuten beraten lassen. In der Piratenpinte sitzen bestimmt welche. Auf dem Weg dorthin verliert der Jungpirat gleich seine gute Meinung vom Piratengewerbe: die Spitzbuben sitzen verschüchtert am Weg oder in der Kneipe beim Grog. Der Grund: allen steckt die unsägliche Furcht vor dem spukenden Seeteufel Le Chuck in den Knochen - besonders dem Piraten, der sich auf dem Leuchter in Sicherheit gebracht hat. Immerhin geben sie dem ahnungslosen Guybrush gute Ratschläge. Aber diese seefesten Jungs schlagen dabei nicht etwa einen lehrerhaften Ton an, sondern produzieren sich vor dem Spieler in köstlich amüsanter Weise. Wie er endlich ein richtiger Pirat wird, erfährt Guybrush im Hinterzimmer beim Anführer. Drei Prüfungen soll Guybrush bestehen, dann wird er in die Gilde der Seeräuber aufgenommen. Zunächst muß er sich einen Degen besorgen und gegen den gefürchtetsten Degenfechter der ganzen Insel antreten. Kein Problem! Danach geht es darum, in die Villa der Gouverneurin zu schleichen und eine Statue zu stehlen. Vorsicht, das Auge des Gesetzes, verkörpert durch den Sheriff, ist nicht weit! Bei der letzten Prüfung soll Guybrush seinen Riecher für Schatztruhen unter Beweis stellen. Irgendwo auf Melee liegt noch ein ungehobener Schatz und wartet auf Entdeckung. Damit ist Guybrush jetzt ein echter Seeräuber. Und gleich unsterblich verliebt. In die Gouverneurin. Doch leider bleibt keine Zeit für Romantik. Der spukende Pirat Le Chuck verschleppt die Schöne nach Monkey Island. Guybrush, der ja gerade sein Piratenexamen bestanden hat, heuert ein paar Kollegen an und nimmt Kurs auf diese Insel. Wie dumm! Seine Crew sprüht nicht gerade vor Arbeitseifer. Sie benimmt sich wie eine Reisegruppe auf Kreuzfahrt -hier ein Drink, da ein Sonnenbad. Dennoch erreicht das Schiff die Insel. Dort wohnen nette Leute. Die früher gefürchteten Kannibalen haben zum Vegetarismus gewechselt - irgendein Gesundheitsapostel überzeugte sie davon, daß Menschenfleisch dem Cholesterinspiegel schaden würde. Als harmlos stellt sich auch der berufsmäßige Schiffbrüchige heraus. Er hat sich in Jahren der Einsamkeit ein Boot gezimmert, bleibt jedoch auf der Insel und harrt seiner ordnungsgemäßen Rettung. Deswegen verleiht er seine seetüchtige Nußschale manchmal. Schöne Voraussetzungen dafür, „Das Geheimnis von Monkey Island" zu lüften. Lucasfilm Games bereicherte sein neuestes Grafik-Adventure nicht nur um witzige Dialoge und Kommentare, sondern auch um neue Einblicke. Bislang sahen Spieler die Szenen immer nur seitlich von vorn. In „Monkey Island" sind die Szenarios in verschiedenen Perspektiven dargestellt - mal sieht der Betrachter das Geschehen von der Seite, aus der Vogelperspektive oder aus einem schrägen Winkel. „Das Geheimnis von Monkey Island" zu lösen, beansprucht Grips und Lachmuskeln. Perfekte Bild- und Tonkulissen, viel Bewegung und amüsante Verwicklungen - bei „Monkey Island" stimmt einfach alles. Für diese Insel ist man gerne reif!

CBO





Fantasy World Dizzy



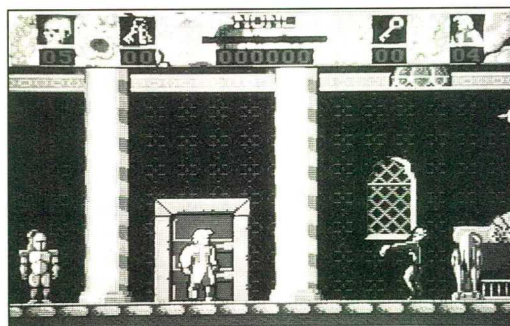


FANTASY WORLD DIZZY ist ein humorvolles Plattformspiel, in dem ein kleines hüpfendes Ei seine Freundin Daisy, die es irgendwo verloren hat, retten muß.

Dies muß es in verschiedenen Welten (Levels) tun, so zum Beispiel in einem alten Schloß, in einem Drachenhort oder sogar durch Erklettern einer magischen Bohnenstange. Das Adventure steckt voller Gefahren, die man mit nur drei verfügbaren „Eiern" überleben muß. Dizzy kann sehr gut springen, so daß eine unendlich weit entfernte Plattform manchmal nur einen Schritt für unseren Helden darstellt. Das Spiel beginnt in einem Schloß, wo zur rechten Seite der Weg von einem Riesen und zur linken von einem lodernden Feuer blockiert ist. Versteckt in einer Ecke, findet Dizzy jedoch eine Kanne voll Wasser, mit der das Feuer zu löschen ist. Mit gefundenen Gegenständen kann man die verschiedenen Puzzles lösen. Hat man eines überwunden, kommt schon das nächste, vielleicht schwierigere, auf einen zu. Goldmünzen und Edelsteine helfen, den Score zu steigern. Da man jedoch nur eine limitierte Zahl an Objekten auf einmal in Besitz halten kann, muß man öfter nachdenken, ob man dieses oder jenes überhaupt mitnimmt, es könnte ja rein gar keinen Nutzen haben. Rätsel zu lösen und den richtigen Weg zu finden, machen jedoch nur einen Teil des Spiels aus. Bösartige Kreaturen, deren Berührung es zu vermeiden gilt, machen Dizzy das Leben schwer. Alligatoren warten nur darauf, ihre Zähne in Dizzy zu versenken, und feuerspeiende Drachen haben Lust auf ein gekochtes Ei. FANTASY WORLD DIZZY ist ein ziemlich gehaltvolles Plattformspiel mit vielen Adventure-Elementen und einer sich stetig ändernden Cartoon-Grafik. Viele unterhaltsame Stunden erwarten Sie.

ddf

Horror Zombies from the Crypt





Schlag Mitternacht flimmerte es auf unserem Redaktionstisch, und ein neues, gefährliches, schleimiges, angsteinflößendes Produkt von Millenium manifestierte sich aus dem Nichts.

Die Handlung dreht sich um die Klischees der amerikanischen B-Movies der 50er Jahre. Man bekommt ein interessantes Adventure, gespickt mit schwarzem Humor und angefüllt mit Zombies, Geistern und anderen Dienern des Dunkels. Die Ruhe von Richter Frederick Valdemars Landsitz wurde plötzlich durch das Auftauchen von Armeen von Untoten gestört, der es umgebende Wald ist voll von Ghouls und Skeletten, die sich dem Haus nähern... Im Garten vergnügen sich Werwölfe und Fledermäuse, und im alten Teil des Hauses befindet sich eine Krypta, in der die Zombies spielen. Nur Sie können den Richter vor etwas Schlimmerem als dem Tod bewahren. Ruhe, Stärke und Geschicklichkeit sind vonnöten, um die alte Ruhe wiederherzustellen. HORROR ZOMBIES ... hat drei enorm große Levels, vollgepackt mit über 600 Screens, höllischen Puzzles und tödlichen Begegnungen. Zusammenfallende Wände, bewegliche Kamine, versteckte Türen und viele geheime Räume gilt es zu erforschen. Die Handlung des Spieles dreht sich um Überleben, Suchen und das Finden von wichtigen Gegenständen. Manche Puzzles sind sehr schwer zu lösen, so daß man immer wieder neu nachgrübeln muß; aber: versuchen Sie, an die einfachen Dinge zu denken! Den zahlreichen Kreaturen ist man nicht schutzlos ausgeliefert, Waffen und sonstige Items können während des Spieles aufgesammelt werden. Haben Sie - wie auch immer - ein Level bewältigt, erhalten Sie ein Paßwort, so daß Sie dieses Level in Zukunft überspringen können. Will man ein Fazit ziehen, so haben Astral, die Programmierer des Spiels, die leicht depressive Atmosphäre der alten Horrorfilme sehr gut getroffen, die Grafik und der Sound tragen hierzu bei.

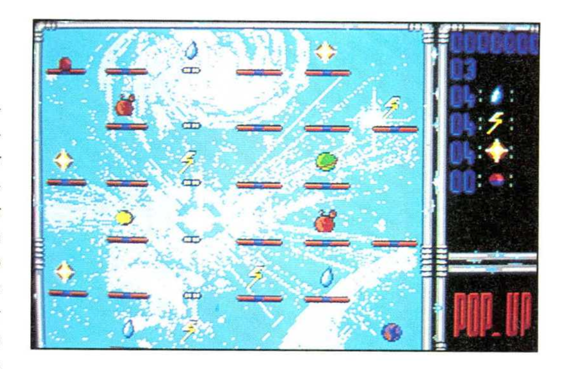
ddf



Inserentenverzeichnis

Pop Up



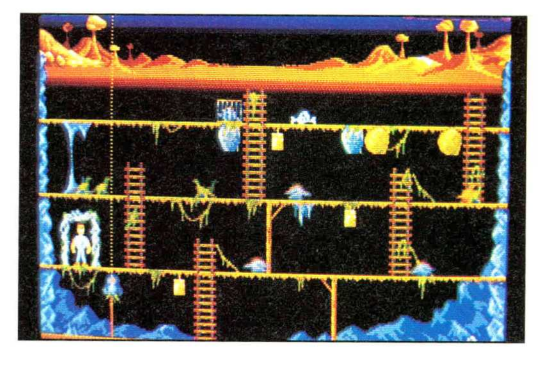


Nach dem phänomenalen Erfolg von „Tetris" und „Arkanoid" besinnen sich immer mehr Hersteller auf simple Spielprinzipien. Daß dabei bisher ebenso schlichte wie schlechte Software ensteht, beweist „Pop Up" auf erschreckende Weise. In Gestalt eines quicklebendigen Gummiballs hat der Spieler fünf Epochen aus jeweils vier Spielabschnitten zu durchhüpfen. Dies geschieht, indem er alle Gegenstände innerhalb eines Bildes aufsammelt und sich anschließend zum Ausgang begibt. So weit - so simpel. Türen, Blockaden und unüberwindbare Feuerwände sollen den Schwierigskeitsgrad erhöhen, sind mit entsprechenden Schlüsseln oder einer Ladung Wasser aber keine ernstzunehmenden Hindernisse. Einzig und allein die schräggestellten Plattformen bereiten Probleme. Einmal berührt, versetzen sie die Kugel in einen unkontrollierbaren Taumel. Da 20 Levels nicht gerade ewig vor den Bildschirm fesseln, haben die Programmierer freundlicherweise an einen Editor gedacht. Der einzige Motivations-Lichtblick! Highscore-Liste, Team-Modus oder Paßwortsystem sucht man vergebens. Jedes Level erinnert auf frappierende Weise an das vorhergehende. Ebenfalls langweilig stimmt die Tatsache, daß es nur einen sinnvollen Ausgang zum nächsten Level gibt. Programmiertechnisch gesehen, ist „Pop Up" nicht gerade ein achtes Weltwunder: die Hintergrundgrafiken in biederer EGA-Qualität wirken verwaschen und lassen nur undeutlich den jeweiligen Zeitabschnitt erkennen. Auch der Sound ist keinen Deut besser. Wer ihn zu lange hört, kratzt vor Ohrenschmerzen die Tapeten von den Wänden. Zugegeben: „Pop Up" ist kinderleicht, leicht spielbar - man braucht auch keine dicken Anleitungsbücher zu wälzen, um das Game zu kapieren. Aber auf die Dauer wird Infogrames neues Geschicklichkeitsspiel zum Schlafmittel.

CBO

A Prehistoric Tale





Bei einem Ausflug in die Vergangenheit macht der Zeitreisende Cronos eine erstaunliche Entdeckung: Vor 150 Millionen Jahren raffen Erdbeben fast alle Saurierbabies dahin. Um ein vorzeitiges Aussterben der Urviecher zu verhindern, beginnt der besorgte Wissenschaftler mit der Evakuierung. In jedem der 50 Levels liegen zahlreiche Sauriereier verstreut, die aufgenommen und zum Teleporter transportiert werden müssen. Junge Reptilien gilt es durch einfaches Überspringen einzufangen, was deren Eltern naheliegenderweise gegen den Strich geht. Eilends auf den Plattformen ausgesetzte Mäuse halten die rachedurstigen Ungetüme eine Zeitlang in Schach, an der richtigen Stelle plazierte Felsbrocken killen sie gar. Zu Beginn muß sich Cronos mit drei Eiern pro Teleporter-Gang begnügen. Das reichhaltige Extra-Arsenal beinhaltet zum Glück eine Vorratsbatterie, die lästiges Hin- und Herlaufen überflüssig macht. Ebenfalls von Nutzen sind Schutzschilde, Waffen und Abkürzungen in höhere Spielstufen. Sollten jetzt immer noch Probleme auftreten, hat Cronos' Freund Jeff die Möglichkeit, ihm unter die Arme zu greifen und in einem der drei Bonuslevels mit auf Punktejagd zu gehen. Schon die an den Haaren herbeigezogene, mit Tippfehlern übersähte Hintergrundgeschichte dieses Geschicklichkeitsspiels aus deutschen Landen läßt nichts Gutes vermuten. Das Schauspiel, das sich dem Betrachter allerdings nach dem Laden auf dem Bildschirm bietet, ist mehr als eine Zumutung. Das mäßig animierte Helden-Sprite läuft, springt und klettert durch eine Anhäufung lieblos zusammengeschusterter Bildschirme. Beim pixelgenauen Springen über die bedenklichen kleinen Monster steht die Kollisionsabfrage beharrlich auf seiten des Computers. Solcherlei Unfairness stört nicht allzu sehr, denn besonders lange hält man es sowieso nicht vor dem Monitor aus. Vom abwechslungslosen Gameplay wird man schnell in den Schlaf gewogen. Kein Wunder, daß das Spiel „Prehistoric Tale" heißt. Das Spielprinzip stammt aus der Urzeit. Anno 1983 mußte man schon auf dem C64 ein Programm namens Dino Eggs über sich ergehen lassen, das diesen verspäteten Nachzügler locker in den Schatten stellt. „Prehistoric Tale" ist ein Flop. Daran ändert auch die kostenlose „Wings of Death"-Demo-Disk in der Verpackung nichts. Was ist los mit Thalion? Normalerweise produziert die Gütersloher Spieleschmiede High Quality-Software. „Prehistoric Tale" scheint wohl ein Ausrutscher zu sein...

CBO

Write On

- ✓ Umfangreiche Handbücher zum Einstieg in die Textverarbeitungen
- ✓ komfortable Installationsprogramme
- ✓ deutsche, automatische Silbentrennung mit erweiterbarer Ausnahmeliste
- ✓ Serienbrieffunktion in Verbindung mit allen gängigen Datenbanken/Adressverwaltungen
- ✓ Fonteditor
- ✓ Textimport von Word+, ASCII
- ✓ Import hochaufgelöster IMG-Grafiken
- ✓ Snapshot Accessory zum Import anderer Grafiken.

Bedienungskonzept

Der Einsteiger erlernt COMPO Software Textverarbeitungssystemen einfach und schnell mit der Maus und Pull-DownMenüs. Dabei lernt man automatisch die schnelle Bedienung aller Funktionen über durchdachte deutsche Tastenkommandos. Als Clou ist es möglich, Befehle, Menüauswahl und Texte nach eigenem Bedarf auf alle Tasten zu legen.

Bedienungsfreundlichkeit

Write On bietet genau das, was man zum Einstieg unter normalen Alltagsbedingungen in einer Textverarbeitung braucht. Keine überflüssigen und den Einsteiger verwirrende Funktionen. Makros und Floskeln, Absatz-/Seitenlayouts, deutsche Silbentrennung, Seitennumerierung, Blockfunktionen, diverse Fonts, Stile und Grafikeinbindungen, werden Sie selbstverständlich in der Write On-Version wiederfinden.
That's Write hat darüber hinaus erweiterte GEM Fenster, und Dialogboxen. Über sogenannte 'Anweisungen' lassen sich komplexeste Büroanwendungen einfach realisieren. That's Write und That's Write PS haben eine internationale Langenscheidt Korrektur und Trennung.

Herausragende Drucktechnologie

Write On und That's Write erlauben die Nutzung druckker interner Schriften (auch proportional in Blocksatz) mit grafischen Schriften und Symbolen in einem Dokument. Der Bildschirm zeigt ständig, was auf das Papier kommt (Vollständiges WYSIWYG). Grafiken sind nicht auf 640x400 Punkte limitiert.

Kundenbetreuung

COMPO Software hat eine spezielle Hotline zur Kundenbetreuung. Dem registriertem Anwender versuchen wir bei Fragen direkt am Telefon zu helfen. Die Hotline ist von Mo-Fr erreichbar. Updates (kleine Verbesserungen im Rahmen der Programmpflege) erhalten Sie i.d.R. gegen frankierten Rückumschlag, Originaldisketten und Bearbeitungsgebühr. Upgrades (Programmerweiterungen, Verbesserungen, ...), werden berechnet.

Textverarbeitung mit System

Wenn Ihr Bedarf an Textverarbeitung und Funktionen steigt, können Sie jederzeit auf That's Write oder auch That's Write PS umsteigen. Ihre Texte werden ohne Veränderung gelesen, die Bedienung bleibt genauso einfach. Profitieren Sie von den erweiterten Funktionen.

Write On in Stichworten:

- einfache Installation und Druckerauswahl durch komfortables Hilfsprogramm, Umfangreiches Handbuch. Läuft unter allen bekannten TOS-Versionen
- Einfachste Bedienung per Maus, Tastatur oder eigener Tastenbelegung durch Makros
  internationale, mehrfach frei belegbare Tastatur (bei Bedarf für jeden Font einzeln) (z.B. é, ê, á, ...)
- zuverlässige, eingebaute Silbentrennung mit ergänzbarem Ausnahmewörterverzeichnis
- Ausdruck von Text- und Grafikmodus in einer Zeile von bis zu 20 Fonts gleichzeitig
- Bildschirmanzeige entspricht Ausdruck (WYSIWYG)
- Flexible Textgestaltung wie beim moderner DTP-Systeme durch Absatz- und Seitenlayouts. Nachträglicher Änderungen sind genauso einfach, wie die Voreinstellungen zu den jeweiligen Aufgabenbereichen
- Grafikeinbindung ohne 640x400 Punkte Limit! — Grafiken skalieren, verschieben und mit maximaler Qualität drucken
  Snapshot-Accessory zur Bildübernahme — der Bildschirminhalt eines beliebigen Programmes kann als Grafik gesichert werden.
- Fonteditor für eigene Zeichen, Signum-Fontkonverter, so können auch Signum-Schriften eingesetzt werden
- viele Druckertreiber für 9-,24-Nadel, SLM804, HP-Desk- und Laserjet
  unterstützt Farb-, Monochrom- und Großbildschirme
- zwei Texte gleichzeitig bearbeitbar
- Makros und Floskeln
- flexibles ASCII-Laden/Sichern. Wahlweise kann der gesamte Text oder ein Block ASCII gesichert werden. Textübernahme von 1st Word/Wordplus. Attribute wie fett, kursiv, hoch/tief, unterstrichen, werden korrekt interpretiert
- Ausschneiden/Einfügen von Blöcken mit 4 unabhängigen Puffern
- Einfüge- und Überschreibmodus auch bei Proportionalschrift
- Einfügen von Datum-kurz, Datum-lang, Zeit, Seitennummer, Folgeseite per Tastendruck
  Hilfsfunktionen wie >Wort klein schreiben<, >Wort groß schreiben<, >Erster Buchstabe groß/Rest klein<
- komfortables Bewegen im Text: Zeichenweise, Wortweise, Zeilenweise, Absatzweise, Bildschirm-Seitenweise, Text-Seitenweise (zur Kontrolle des Seitenumbruchs), Text Anfang/Ende, Block Anfang/Ende, Seitenanfang,... 10 Marken zum schnellen Anspringen von Textpassagen
- Serienbrieffunktion in Verbindung mit allen gängigen Datenbanken und Adressverwaltungen
- Druckausgabe auf Druckerport, RS232 oder Datei
  Bilder können beim Drucken auch weggelassen werden (als Probeausdruck zum Beispiel)

Preis 148,– DM
plus Porto und Versand 6,– DM
Endpreis 154,– DM

Preis ist ein unverbindlich empfohlener Verkaufspreis

BESTELL - COUPON

Heim Verlag
Heidelberger Landstraße 194
6100 Darmstadt-Eberstadt
Telefon 06151/56057
Telefax 06151/56059

Bitte senden Sie mir: ____ Write ON à **148,– DM**
zuzüglich Versandkosten DM 6,- (Ausland DM 10,-) unabhängig von der bestellten Stückzahl
☐ per Nachnahme ☐ Verrechnungsscheck liegt bei

Name, Vorname ____________________
Straße, Hausnr. ____________________
PLZ, Ort ____________________
Oder benutzen Sie die eingeheftete Bestellkarte

In Österreich:
RRR EDV GmbH
Dr. Srumpfstraße 118
A-6020 Innsbruck

In der Schweiz:
Data Trade AG
Landstr. 1
CH - 5415 Rieden-Baden



Signum!-Zeichensätze in Omikron.BASIC

Die Ästheten unter den ST-Anwendern blicken schon immer neidisch auf gewisse andere Fabrikate, die von vornherein mit hübsch proportionalen Zeichensätzen arbeiten. Zwar ist auch der ST von Natur aus dazu geeignet, doch die meisten Hobbyprogrammierer verzichten darauf, sich dafür durch den GEM/VDI-Dschungel zu kämpfen. Für einfache Anwendungen genügt aber auch die hier vorgestellte kurze Omikron.BASIC-Routine. Sie verwendet Signum!-Bildschirm-Fonts (die als Public Domain reichlich zur Verfügung stehen) und kommt ohne GEMLIB aus.

Die Routine *PLOAD()* lädt eine/mehrere E24-Dateien in den Speicher und patcht sie für die Ausgabe mit *BITBLT*. Jeder Zeichensatz muß dabei mit einem Namen gekennzeichnet werden, so daß später mit der Anweisung *FONT=Name* ein beliebiger Zeichensatz ausgewählt werden kann. Am Ende des Programms sollte der Speicher mit *FRE(Name)* wieder freigegeben werden.

Die Routine *PTEXT (X,Y, Text$)* wird genau wie die BASIC-Anweisung *TEXT* eingesetzt - mit dem Unterschied, daß die Variable X danach auf das Ende des ausgegebenen Textes zeigt, so daß man die nächste freie Position auf dem Bildschirm kennt.

Arno Maurer, W-7400 Tübingen 1

```
 1: Pload("SWISS,E24",Swiss)
 2: INPUT "  Probetext? ";T$
 3: Font=Swiss
 4: CLS
 5: Ptext(90,90,T$)
 6: FRE(Swiss)
 7: END
 8: '
 9: DEF PROC Pload(File$,R Font)
10:   LOCAL A,I
11:   OPEN "I",1,File$
12:   Font=MEMORY(LOF(1))
13:   CLOSE 1
14:   BLOAD File$,Font
15:   FOR I=1 TO 127
16:     A=Font+652+LPEEK(Font+I*4+140)
17:     POKE Font+I,PEEK(A)
18:     POKE A+3,PEEK(A+1)
19:     WPOKE A,PEEK(A+2)
20:     POKE A+2,0
21:   NEXT I
22: RETURN
23: '
24: DEF PROC Ptext(X,Y,Text$)
25:   LOCAL A,B,I,S,Z
26:   FOR I=1 TO LEN(Text$)
27:     Z=ASC(MID$(Text$,I,1))
28:     S=INSTR("äöüÄÖÜ§ß",CHR$(Z))
29:     IF Z=32 THEN Z=0
30:     IF S THEN
31:       Z=ASC(MID$("][@}{\ "+CHR$(127),S,1))
32:     ENDIF
33:     IF Z>127 OR Z=0 THEN
34:       X=X+8
35:     ELSE
36:       S=WPEEK(A)
37:       WPOKE A,2
38:       B=WPEEK(A+2)
39:       BITBLT A TO X,Y-17+PEEK(Font+Z),B,
          WPEEK(A+4),7
40:       WPOKE A,S
41:       X=X+B
42:     ENDIF
43:   NEXT I
44: RETURN
```

Listing 1: Signum!-Zeichensätze in Omikron.BASIC

Selbstdefinierte Linienmuster in GFA-BASIC

Der GFA-BASIC-Befehl *DEFLINE [stil], [dicke], [anfang_s, end_s]* erlaubt bekanntlich neben der Wahl von Liniendicke, Anfang- und Endsymbol auch das Festlegen des Linienstils für die Befehle *LINE*, *BOX* usw. Hier gibt es die Möglichkeit, zusätzlich zu den 6 bzw. 7 vordefinierten Linienarten auch selbstdefinierte Muster zu benutzen - wenn man weiß, wie. Alle mir bekannten Angaben dazu (einschl. orig. GFA-Handbuch, allerdings 1. Aufl. vom Mai '88) sind entweder unvollständig oder schlicht falsch.

Der Knackpunkt ist, daß das gewünschte Linienmuster als 16-Bit-Wert im Zweierkomplement angegeben werden muß (das scheint komplizierter zu sein, als es tatsächlich ist)! Beispielsweise erhält man eine Linie aus je zwei gesetzten und zwei nicht gesetzten Punkten durch den Befehl

```
DEFLINE 1+NOT
  &X1100110011001100,1,0,0
```

bzw., wenn das Muster in einer Variablen vorliegt, mit

```
stil%=&X1100110011001100
DEFLINE 1+NOT stil%,1,0,0
```

Die Übergabe des Wertes für das Muster als negative Zahl (wie es in den meisten Veröffentlichungen dazu immerhin heißt) ergibt sich so von selbst - diesmal jedoch mit dem erwarteten Ergebnis. Dabei ist die Reihenfolge der Operatoren wichtig: ... *NOT wert+1* ... liefert ein falsches Ergebnis. Es geht aber ... *(NOT wert)+1*.

Volker Goreth, W-4150 Krefeld 1

Kleiner Zeichensatz mit PRINT

Wer fand es noch nicht ärgerlich, daß man mit dem *PRINT*-Befehl immer nur den normalen Zeichensatz benutzen kann? Dabei gäbe es so viele Anwendungen, bei denen man den kleinen Zeichensatz (8*8) wesentlich besser einsetzen könnte. Es wäre sicherlich sehr interessant, wenn man statt 25 nunmehr 50 Zeilen auf dem Monitor unterbringen könnte! Tabellen, die normalerweise auf zwei Seiten aufgeteilt werden müßten, passen nun auf einmal auf eine Seite; Listings und Texte werden wesentlich übersichtlicher. Die einzige Abhilfe für dieses Problem war bislang der *TEXT*-Befehl, der jedoch nicht gerade vor Schnelligkeit strotzte, sondern eher wesentlich langsamer als das *PRINT*-Äquivalent war. So gaben sich die meisten Programmierer damit zufrieden, daß es nicht anders geht, und teilten ihre Tabellen in mehrere Teile auf. Diesem Problem kann jedoch abgeholfen werden mit einem recht kurzen Listing, das alle folgenden *PRINT*-Anweisungen im kleinen 8*8-Zeichensatz ausgeben läßt. Dabei sollte man lediglich beachten, daß man nach dem Umschalten auf eine andere Font-Größe immer den Bildschirm löscht, da das Betriebssystem sonst mit der Darstellung der Fonts durcheinanderkommt. Ansonsten steht dem großflächigen Aufbau von Tabellen oder Datenaufkommen jedweder Art nichts mehr im Wege. Viel Spaß mit der neuen, "großen" Bildschirmauflösung!

S. Just, W-6936 Haag / MP

```
 1: font_init               !Adressen suchen
 2: text(small_font%)       !kleiner Font
 3: CLS                     !Bildschirm löschen
 4: FOR i&=1 TO 1000        !...mal ansehen
 5:   PRINT "Test! ";       !
 6: NEXT i&                 !...und raus
 7: text(big_font%)         !großer Font
 8: CLS                     !Bildschirm löschen
 9: FOR i&=1 TO 1000        !...mal ansehen
10:   PRINT "Test! ";       !
11: NEXT i&                 !...und raus
12: END                     !...Ente
13: '
14: PROCEDURE font_init    !Adressen finden
15:   LOCAL find_match$,find_a%,find_m.adr%
16:   ' 0=6*8, 1=8*8, 2=8*16
17:   find_match$=MKI$(&HA000)+MKI$(&H2009)+
            MKI$(&H4E75)
18:   find_a%=VARPTR(find_match$)
19:   find_m.adr%=C:find_a%()
20:   big_font%=LPEEK(find_m.adr%+8)    !groß
21:   small_font%=LPEEK(find_m.adr%+4) !klein
22: RETURN
23: PROCEDURE text(fnt_adr%)    !Umschalten
24:   IF fnt_adr%>0
25:     {INTIN}=fnt_adr%
26:     INT{CONTRL+2}=0
27:     INT{CONTRL+6}=2
28:     INT{CONTRL+10}=102
29:     VDISYS 5
30:   ELSE
31:     ALERT 3,"Font-Adresse|unbekannt!",1,"RETURN",d%
32:   ENDIF
33: RETURN
```

Listing 2: So schaltet man den System-Font in GFA-BASIC um.

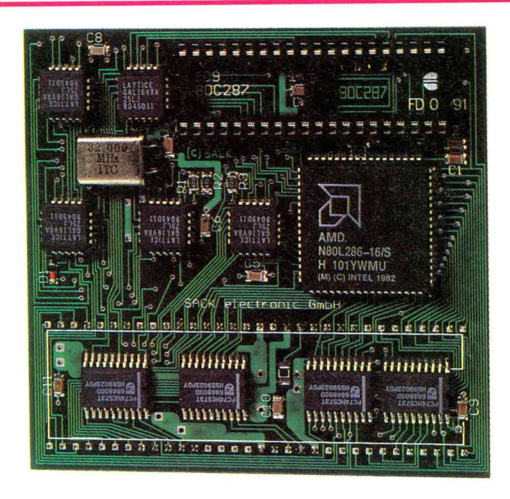












Für die Zukunft gerüstet:

16 MHz	AT-*Speed C16* verfügt über einen mit 16 MHz getakteten 80286-Prozessor.
Norton 8.2	Es wird ein Norton-Faktor von 8.2 erreicht.
Steckplatz für Co-Prozessor	AT-*Speed C16* besitzt einen Steckplatz für einen mathematischen 80C287-Co-Prozessor. Dieser Co-Prozessor kann auch später nachgerüstet werden. Der 80C287-Co-Prozessor wird im DOS-Bereich von vielen Programmiersprachen, Tabellenkalkulationen, CAD-Programmen etc. vorbildlich unterstützt. Weit über 100% Geschwindigkeitszuwachs werden durch diese Option möglich. Die Anzahl der Programme, die ohne einen Co-Prozessor nicht lauffähig sind, nimmt ständig zu.
EGA/VGA-mono*	Hercules, CGA, Olivetti, Tandy 16 Farben, ATT 400
DR DOS 5.0	Das Betriebssystem ist im Lieferumfang enthalten.

*Soweit es die Fähigkeiten des ATARI ST/Mega ST erlauben.
Technische Änderungen vorbehalten

Vertrieb weltweit:

Heim Verlag

Heidelberger Landstraße 194
6100 Darmstadt-Eberstadt
Telefon 0 61 51 - 5 60 57

In Deutschland: Über 400 ATARI-Händler informieren, beraten, bauen ein, betreuen

Österreich: Darius
Inh. K. Hebein
Hartlebengasse 1-17/55
A-1220 Wien

Schweiz:Data Trade AG
Landstr. 1
CH-5415 Rieden-Baden

für alle anderen Länder:

COMPO
SOFTWARE GmbH
Postfach 1051
D-5540 Prüm (FRG)
Tel.: 0 65 51 / 62 66

Händler:

Rufen Sie an. Wir nennen Ihnen gern Ihren Händler

Die Privaten sind ab sofort kostenlos!



ST ohne Laufwerk, gegen Gebot. (keine Rollschuhe, hab schon 3) Bin nach 8 in der Pinte, dort anrufen.

Festplatte, 4 Jahre alt, kaum gebraucht, mit Rollen (daher nicht mehr ganz fest) im Tausch gegen ATARI ST 1040. Bei Clausen Nachricht hinterlassen.

Rollschuhe, 20MB RENN, mit Verwechselplattenlaufwerk, 900.- DM VHB.

HARDWARE

ATARI ST , 1 Jahr alt, für umsonst oder gegen 1 Paar Rollschuhe. Tagsüber unter dieser Nr. nicht zu erreichen, besser abends.

TT, 50 Mhz , kein 1 Jahr alt, nicht für alles Geld der Welt, oder gegen ein Paar Rollschuhe. Zuhause anrufen.

Kleinanzeigen-Auftrag ST-Computer

Bitte veröffentlichen Sie für mich folgende Kleinanzeige in der angekreuzten Rubrik:

○ **Biete an** ○ **Hardware** ○ **Software** ○ **Ich suche** ○ **Kontakte** ○ **Tausch** ○ **Verschiedenes**

30 Buchstaben je Standardzeile - incl. Satzzeichen und Wortzwischenräume.
Groß- und Kleinbuchstaben verwenden, fettgedruckte Wörter unterstreichen.

Bearbeitung bei gewerblichen Kleinanzeigen nur gegen Vorausscheck über den entsprechenden Betrag (keine Überweisung).
Der Verlag behält sich das Recht vor, die Veröffentlichung zu verweigern, wenn dies erforgerlich erscheint!
Bei Angeboten: Ich bestätige, daß ich alle Rechte an den angebotenen Sachen besitze.

○ Private Kleinanzeige (kostenlos)
○ Gewerbliche Kleinanzeige für 12.- DM je Druckzeile
○ Scheck über DM _____ ist beigefügt

An: Heim Verlag
"ST-Computer-Kleinanzeige"
Heidelberger Landstr. 194
6100 Darmstadt-Eberstadt
Telefon 0 61 51 - 5 60 57

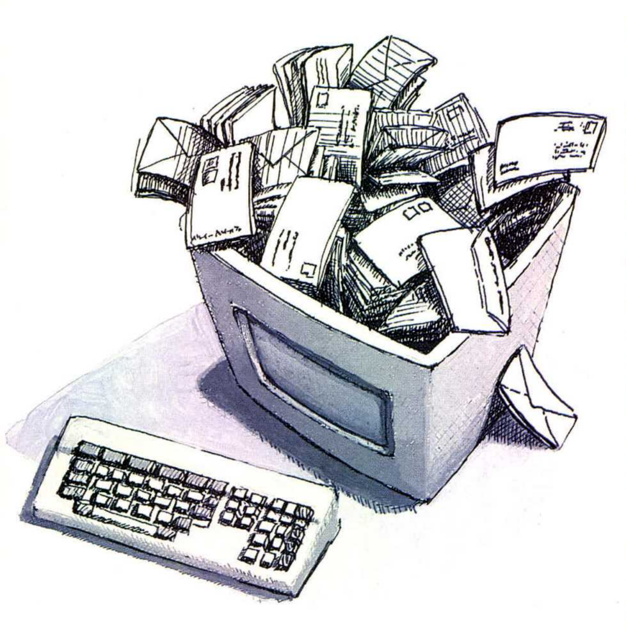
Meine Adresse:

Name: ____________________

Str. ____________________

Wohnort: ____________________

Datum,Unterschrift ____________________



Ein Wort in eigener Sache

In den Jahren, die unsere Zeitschrift existiert, haben wir immer wieder versucht, durch die Beantwortung der bei uns eingehenden Briefe ein wenig Licht in das Dunkel zu bringen, das bei der Arbeit mit dem ATARI ST schon so manch einen aus der Fassung bringen konnte - eine Tatsache, die nicht nur Ihnen, verehrter Leser, sondern auch uns oft genug zu schaffen machte. Nichtsdestotrotz haben wir uns bemüht, die Probleme zu lösen und diverse Leserbriefe zu veröffentlichen, da wir der Meinung waren, daß die jeweilige Thematik auch einen größeren Leserkreis interessieren könnte. Trotzdem gibt es immer wieder Briefe, die wir nicht beantworten können oder dürfen. Damit Sie nicht allzusehr enttäuscht zu sein brauchen oder keine Antwort erhalten, möchten wir Sie bitten, sich an folgende Spielregeln zu halten, die sich aus unserer Erfahrung ergeben haben. Fällt Ihr Brief nicht unter die folgenden Kriterien, hat er gute Chancen, positiv beantwortet oder wenigstens als Hilferuf an unsere Leserschaft gedruckt zu werden.

1. Leider gehen immer wieder Briefe mit dem Wunsch ein, ein Produkt für diesen oder jenen Anwendungsfall vorzuschlagen, verschiedene Produkte bezüglich der Vor- und Nachteile gegeneinander abzuwägen und zu bewerten. Es ist uns aus Wettbewerbsgründen nicht erlaubt, ein bestimmtes Produkt zu favorisieren, selbst wenn wir das eine oder andere in der Redaktion überzeugt einsetzen. Wir können Sie in diesem Fall ausschließlich auf die von uns möglichst objektiven Tests und eventuell anstehende Fachmessen hinweisen. Bedenken Sie bitte, daß auch wir nicht jede Textverarbeitung, jedes Malprogramm und so weiter kennen und bestimmte Produkte dadurch in das Abseits drängen würden.

2. Oft erreichen uns Briefe, die sich positiv oder auch negativ über bestimmte Händler, Softwarehäuser oder deren Produkte auslassen. Sicherlich interessieren uns solche Bemerkungen. Bitte haben Sie aber Verständnis, daß wir weder Lob noch Tadel abdrucken dürfen, da diese Aussagen meist subjektiv sind. Anders sieht die Sache beispielsweise bei Gerichtsurteilen aus, die Sie, verehrte(r) Leser(in), erfochten haben.

3. Aufgrund der Vielzahl an Briefen, die uns täglich erreichen, sind wir leider nicht in der Lage, Programmfehler anhand von Listings oder ähnlichem zu korrigieren. Dennoch sollte ein Problem möglichst detailliert beschrieben sein, denn Ferndiagnosen sind prinzipiell sehr schwer, jedoch mit genauerer Angabe der Symptome eventuell durchführbar.

4. Von Zeit zu Zeit erreichen uns Briefe mit der Bitte, die Adresse des Lesers zwecks allgemeiner Kontaktaufnahme zu veröffentlichen. Würden wir dies in die Tat umsetzen, würde sich der Umfang des anderen redaktionellen Teils beträchtlich verkleinern. Ausnahmen stellen Leser in fernen Ländern dar, für die eine Kontaktaufnahme im eigenen Land recht schwierig ist.

Zum Schluß sollen ein paar Tips eventuell voreilig geschriebene Briefe verhindern.

1. Wenn Sie ein Problem bezüglich einer bestimmten Problematik haben oder an einem bestimmten Produkt interessiert sind, finden Sie interessante Artikel darüber eventuell in vorhergehenden Ausgaben userer Zeitschrift. Zur Auswahl eignet sich das Jahresinhaltsverzeichnis besonders gut, das immer am Jahresende in der ST Computer abgedruckt wird.

2. Sollten die Probleme mit der Handhabung eines Produktes zu tun haben, wenden Sie sich zunächst an Ihren Händler und über diesen an den Distributor beziehungsweise an das Software-Haus. Die Wahrscheinlichkeit, daß Ihnen das Software-Haus weiterhelfen kann, ist um ein Vielfaches höher als die, daß wir Ihnen helfen können.

3. Lesen Sie aufmerksam die Leserbrief-Seite. Viele Fragen wiederholen sich immer wieder, obwohl wir bestimmte Probleme schon mehrfach angesprochen haben.

Dokumentationen zum ST/TT

Ich bin zur Zeit Student an der Fachhochschule Saarbrücken im Fachbereich Elektrotechnik. Im Rahmen einer Studienarbeit befasse ich mich mit Betriebssystemen von Computern. Für diese Arbeit benötige ich auch Dokumentationen über die Betriebssysteme der Atari ST- und TT-Serie. Ich wende mich an Sie, da mir die einschlägige Literatur nicht genügend Informationen liefert. Wie und wo kann ich solche Unterlagen erhalten (abgesehen von Atari Deutschland)?

Jörg Paulus, W-6601 Heusweiler

Red.: Die Firma Atari dürfte sicher in der Lage sein, Ihnen diese Informationen zu geben. Sollten Sie dabei auf Probleme stoßen, böten sich immerhin einige Bücher an: u.a. das *Profibuch* aus dem Sybex-Verlag, *Scheibenkleister II* von MAXON, *Atari ST Intern* von Data Becker, *Vom Anfänger zum GEM-Profi* aus dem Hüthig-Verlag oder auch *Das TOS-Listing* aus dem Heise-Verlag an, aus denen Sie sicher auch die meisten benötigten Informationen entnehmen können.

*

Grafikprogrammierung

Schon seit langem programmiere ich mit meinem Mega ST4 und habe ein Problem, das sicher auch schon einige andere Leser hatten/haben. Ich habe eine kleine Grafikanimation programmiert und würde gerne, während diese abläuft, noch andere Grafiken auf den Bildschirm bringen oder einen Sound abspielen lassen. GFA-BASIC EWS 3.5 bietet, soweit ich weiß, aber leider keine Funktion zum gleichzeitigen Durchlaufen mehrerer Schleifen. Da aber fast alle Computerspiele solche Effekte haben und der ST zum Umschalten der Systemzeit auch keine Programmunterbrechung benötigt, muß es irgendwie möglich sein, in einer sehr hohen Frequenz zwischen zwei Schleifen hin- und herzuspringen. Wie programmiert man ein solches „Multitasking"?

Jan Dünnweber, W-8044 Lohhof

Red.: In GFA-BASIC läßt sich ein Quasi-Multitasking mit den Befehlen EVERY...AFTER oder EVERY x bzw. EVERY TICKS programmieren. Mit diesen Anweisungen veranlassen Sie den Interpreter/Compiler, nach einer bestimmten Zeiteinheit zu einer bestimmten Prozedur zu springen und diese abzuarbeiten (bzw. jeweils einen Befehl aus der Prozedur). Dadurch dürfte es relativ problemlos möglich sein, auch in GFA-BASIC Quasi-Multitasking-Prozesse zum Laufen zu bekommen.

*

Kalter Start

Ich besitze einen 1040 STE und habe mir nun eine Speichererweiterung auf 4 MB geleistet. Jetzt aber tritt folgender Effekt auf: Wenn ich eine RAM-Disk anmelde oder ein resetfestes Programm lade und den Rechner für 6 Sekunden ab- und dann wieder einschalte, ist die RAM-Disk noch da, oder das Programm läuft weiter wie nach einem Reset. Erst nach zirka 8 bis 10 Sekunden Abschaltpause erfolgt ein Kaltstart; wenn kein Monitor angeschlossen ist, dauert's noch länger. Auf den alten SIMM-Modulen waren je nur 2 Kondensatoren, auf dem neuen ist je Chip ein Kondensator aufgelötet. Nun die Frage: Wie kann ich den Rechner dazu bringen, auch wie früher nach 1 Sekunde schon kaltzustarten?

A. Kromat, O-1166 Berlin

Red.: Ein Kaltstart ist in diesem Fall mit der Tastenkombination Control-Alt-Right Shift-Delete (ab TOS 1.04, also auch beim STE möglich) problemlos zu bewerkstelligen. Daß die Daten noch im Speicher sind, liegt daran, daß bei kurzzeitigem Ausschalten nicht das gesamte RAM gelöscht wird.

Toner für Atari-Laser

Seit zwei Monaten besitze ich den neuen Atari-Laserdrucker SLM 605. Durch Zufall las ich einen Test über drei baugleiche Drucker in der c't 2/91. Meiner Meinung nach haben diese drei Drucker und der Atari-Laser das gleiche Druckwerk. Wenn das so ist, müßte man doch auch die Verbrauchsmaterialien wie Trommel, Toner usw. von den Anbietern dieser Geräte benutzen können. Bei einer Übereinstimmung meiner Beobachtungen wäre dies sicherlich für alle Atari-Laser-Besitzer interessant, da doch erhebliche Preisunterschiede bestehen.

Claus-Peter Mühlhausen, W-2394 Saltrup

Red.: Alle Verbrauchsmaterialien wie Toner, Trommel usw. können auch von anderen Herstellern gekauft werden. Das Druckwerk des SLM 605 ist baugleich mit den Geräten Toshiba Pagelaser 6, Schneider Laserprinter, Vobis LP 66 und C.Itoh CI-6. Lediglich beim Gerät von Schneider könnte es Probleme geben, da hier das Druckwerk leicht geändert worden ist. Toner kostet für den Lasersprinter DM 79,50, für LP 66 und CI-6 DM 29,83 (Abgabe im Sechserpack) und für den Pagelaser 6 DM 101,46. Auch die Preise für die Trommeln sind recht unterschiedlich. Lasersprinter: DM 298,-, LP 66/CI-6: DM 198,-, Pagelaser 6: DM 290,70. Tonerkassetten sind auch bereits von Pelikan zu einem günstigen Preis erhältlich.

*

KAOS-TOS ST-Magazin

Im ST-Magazin, Ausgabe 1 und 2, ist im Anbieterverzeichnis im Zusammenhang mit dem KAOS-TOS Ihre Anschrift erwähnt. Meine Anfrage an Sie ist, ob Sie dieses Betriebssystem inzwischen anbieten.

E. Fricke, W-3060 Stadthagen

Red.: Viele Anfragen erreichten uns zu diesem Thema, was daher rühren mag, daß das Anbieterverzeichnis dieser Ausgabe des ST-Magazins nicht gerade ausführlich ist. Wie Sie auf Seite 39 der Ausgabe 2 des ST-Magazins lesen können, wird die Firma MAXON als Anbieter der Beschleunigerkarte „Board 20" genannt. Das KAOS-TOS wird nicht von uns vertrieben und auch weiterhin nicht werden. Uns ist bislang auch keine andere Firma bekannt, die das KAOS-TOS vertreiben wird. Sollte sich hier etwas ändern, werden wir natürlich an geeigneter Stelle darauf hinweisen.

*

gdat

Als wir diese Woche die Firma gdat telefonisch kontaktieren wollten, mußten wir feststellen, daß der Anschluß nicht mehr besteht. Auf Anfrage wurde uns von der Fernmeldeauskunft mitgeteilt, daß ein Anschluß der Firma gdat in Bielefeld seit Oktober '90 nicht mehr existent ist. Nach Durchsicht Ihrer Zeitschrift auf einen Verweis auf Verbleib der Firma ist mir aufgefallen, daß gdat in Ihrer Zeitschrift nicht mehr inseriert. Da wir auf die Lieferungen und die Betreuung durch die Firma im Falle von defekten ROM-Modulen usw. angewiesen sind, bitten wir Sie um Mithilfe. Gibt es gdat noch bzw. hat diese Firma ihren Sitz verlegt? Hat eine andere Firma den Vertrieb, die Software-Pflege bzw. Betreuung übernommen? An wen sollte man sich wenden? Ist überhaupt noch ein Bezug von ROM-Modulen möglich?

M Schweikart, W-7482 Kraucherwies-Ablach

Red.: Die Firma gdat ist nicht mehr existent, auch nicht an anderer Stelle. Einen Teil der Produkte hat die Firma Eickmann Computer in Frankfurt übernommen, an die Sie sich in diesem Fall wenden sollten.

*

Wordplus-Fehler

Als Benutzer des Textverarbeitungsprogramms Wordplus mußte ich leider die schmerzliche Erfahrung machen, daß die Benutzung des Apostrophs (') zur Hervorhebung von bestimmten Wörtern zum Programmabsturz führen kann. Folgende Voraussetzungen müssen für einen Programmabsturz erfüllt sein:

1. Die „Trennhilfe"-Funktion (?) muß eingeschaltet sein.
2. Das letzte Wort einer Zeile muß mit einem Apostroph (') beginnen, außerdem muß die Trennhilfe versuchen, dieses Wort zu trennen.

Danach werden Sie sich sicher freuen, den soeben bearbeiteten Text noch vorher auf Diskette oder Platte abgespeichert zu haben, denn anstatt Ihres Textes sehen Sie nun drei nette Bömbchen auf dem Bildschirm.

Regel 1: Benutzen Sie nie ein Apostroph anstelle von Anführungsstrichen, um Textstellen hervorzuheben.
Regel 2: Läßt sich Regel 1 nicht einhalten, schalten Sie die Trennhilfe ab!
Regel 3: Benutzen Sie Wordplus Version 1.89!

Winfried Radner, W-7131 Wurmberg

Red.: Alle Benutzer von Wordplus seien hiermit auf den Fehler hingewiesen und wissen nun, wie man den Fehler vermeiden kann. Solange keine neue Version von Atari erhältlich ist, müssen Sie (leider) mit dem Fehler leben. Laut Auskunft von Atari ist allerdings bald die Version 3.15a erhältlich, aus der dieser Fehler (und natürlich noch einige andere) entfernt wurden. Außerdem wird die neue Version auch problemlos auf dem TT laufen. Sobald die neue Version 3.15a offiziell von Atari verfügbar ist, wird sie für alle registrierten Anwender angeboten. Update-Bedingungen sind uns derzeit leider noch nicht bekannt.

Farbfernseher

Ich möchte meinen Atari 1040 ST an ein Farbfernsehgerät anschließen. Ich bitte um Übersendung eines Anschlußplans. Vielen Dank für Ihre Mühe!

Jochen Seidel, W-7210 Rottweil 1

Red.: Immer wieder erhalten wir Anfragen, wie man dieses oder jenes Gerät anschließen kann. Natürlich können auch wir nicht jedes Gerät kennen oder gar besitzen und haben daher auch nicht alle Anschlußbelegungen vorrätig. Wenn Ihr Fernseher einen SCART-Anschluß besitzt, stellt der Anschluß kein Problem dar, denn dann benötigen Sie lediglich ein fertig konfektioniertes Kabel, das Sie bei jedem Computer-Händler erhalten können. Ist Ihr Fernsehgerät jedoch nur mit einem normalen Antennenanschluß ausgestattet, müssen Sie sich einen sogenannten *HF-Modulator* basteln oder kaufen. In diesem Fall wenden Sie sich bitte an den Schaltungsdienst Lange in Berlin, der Ihnen bei allen Problemen mit Schaltungen gerne weiterhelfen wird.

*

Druckertreiber

Ich besitze einen Mega ST2 und das Textprogramm WordPerfect Version 4.1 deutsch. Dazu habe ich mir den IBM-Matrixdrucker *IBM 2205-001* angeschafft. Leider, und dies stellte ich erst fest, als ich einen Text ausdrucken wollte, wird bei WordPerfect kein Druckertreiber für dieses Gerät mitgeliefert. Bitte geben Sie mir einen Tip, wie ich den Druckertreiber für den IBM-Matrixdrucker 2205-001 in das Textprogramm WordPerfect 4.1 deutsch installieren kann.

B. Linke, W-4000 Düsseldorf 13

Red.: Den Druckertreiber bzw. Tips zur Installation können Sie aller Wahrscheinlichkeit nach direkt bei WordPerfect Software GmbH, Frankfurter Straße 33-35, W-6236 Eschborn.





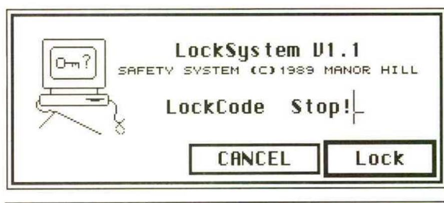
Sie sind beliebt bei jung und alt. Sie verbreiten überall gute Laune. Sie sind innovativ, kreativ und überdies völlig uneigennützig. Und sie erscheinen jeden Monat in der PD-NEWS – die besten Programme aus der PD-Serie der ST-Computer.

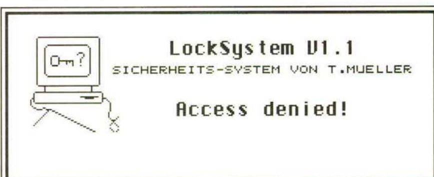
Hände weg!

Beschleicht Sie manchmal so ein ungutes Gefühl, wenn Sie Ihren eingeschalteten Atari während der Arbeit einmal einen Moment aus den Augen lassen müssen, um frischen Kaffee zu kochen oder eine Tüte Chips zu besorgen? Wahrscheinlich wird gerade in diesem Augenblick jemand das Zimmer betreten und an dem Rechner herumspielen (z. B. neugierige Eltern, Kinder, Geschwister, Freunde/innen), mit dem Ergebnis, daß hinterher irgendetwas gelöscht oder unauffindbar ist. Um solche unangenehmen Überraschungen zu vermeiden, sollte man das LOCK-Accessory von Thomas Müller im Wurzelverzeichnis seiner Boot-Diskette installieren.

Nach Aufruf des Accessorys läßt sich ein beliebiges sechsstelliges Paßwort eingeben, das Eingaben über Tastatur oder Maus unmöglich macht. Lediglich die Zauberkombination Alternate/Help funktioniert (leider oder zum Glück?) noch. Um diese „Kindersicherung" wieder zu lösen, wird das Paßwort blind - die Buchstaben werden dabei nicht angezeigt - erneut eingetippt. Das Erraten läßt sich durch Verwendung von Control-Kombinationen erheblich erschweren. Hat man die richtigen Buchstaben allerdings selbst vergessen, bleibt nur noch ein Griff zum Reset-Knopf. - Für jeden, der noch einen Accessory-Platz frei hat, eine prima Sache.

thl





Lock
ST-PD 356

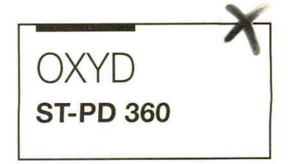
Schwierigkeiten mit OXYD?

OXYD (PD 360) enthält so mancherlei Knobelei. Auch in den ersten 10 PD-Landschaften geht es ganz schön zur Sache. Für all' jene, die noch nicht den richtigen Dreh zur Lösung dieser Knobeleien gefunden haben, haben wir an dieser Stelle ein paar interessante Tips.

Die erste Landschaft, in der die meisten Spieler Schwierigkeiten haben, ist mit der Nummer 5. In deren oberem Teil befinden sich zwei blinkende Punkte. Die den Bildschirm umrahmenden Steine mit dem Loch in der Mitte lassen sich durch Anstoßen verschieben. Einer dieser Steine braucht bloß auf den linken blinkenden Punkt geschoben zu werden. Dadurch öffnet sich im unteren Teil der Landschaft eine Tür, hinter der sich ein Schalterstein befindet. Mit Hilfe eines Regenschirms (muß vorher aktiviert werden!) kann man dorthinfliegen und diesen betätigen. Damit öffnet sich wiederum im oberen Teil der Landschaft eine Tür. Auf diese Weise kann man nacheinander alle Türen öffnen.

Vielen ist auch die Funktionsweise der Meditationslandschaften unklar. Das Prinzip ist aber äußerst einfach: Sie steuern alle Kugeln gleichzeitig(!) mit der Maus. Sie müssen lediglich in jede Mulde eine Kugel manövrieren. Bei der Landschaft 10 sollte man die Kugeln ruhig in die drehenden schwarzen Löcher hineinfallen lassen; fällt eine Kugel in das schwarze Loch, so erscheint sie anschließend wieder auf dem anderen Brett. Allerdings sollten Sie keine allzu hektischen Bewegungen mit der Maus veranstalten, da sonst die Kugeln von den Brettern fallen. Versuchen Sie einmal eine Kugel nach der anderen vorsichtig in die Mulden abzudrängen. Sie werden sehen, wie einfach das eigentlich ist!

Meinolf Schneider





Midi mal anders!



Ein schneller Blick um die Ecke. Da ist einer! Ein ahnungsloses Opfer, also nichts wie hinterher. Der andere ist ganz schön fit. Er rast mit halsbrecherischer Geschwindigkeit um alle Ecken, und um ein Haar hätte er es geschafft, mich abzuhängen. Aber ich bleib' dran und warte auf meine Chance. Er verschwindet um eine Ecke, ich reiße den Knüppel rum.

Mist! Da ist noch einer. Die beiden liefern sich ein Duell. Mein Opfer scheint schon angeschlagen zu sein. Er weicht zurück. Zu spät entdeckt er mich. Ein Gnadenschuß, und er ist erledigt. Der Dritte ist jetzt sicher sauer. Ich habe ihm ja seine Beute vor der Nase weggeschnappt. Und gleich wird er sich noch viel mehr ärgern, denn er wird mein nächstes Opfer.

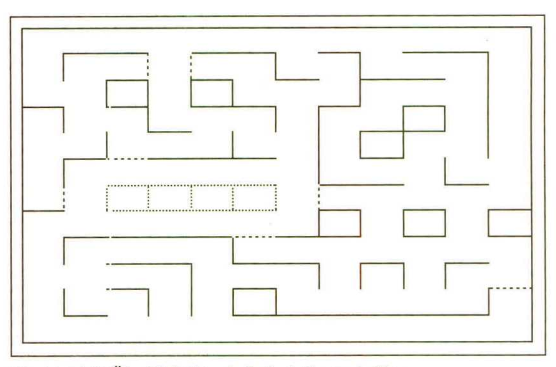
Er schießt. Ich weiche geschickt nach hinten um die Ecke aus. Vorbei! Ein reflektierender Schuß um die Ecke, ich höre, wie er trifft. Ich stürme nach vorne. Da ist er mit seinem breiten Grinsen im Gesicht. Doch das vergeht ihm gleich, denn auch mein nächster Schuß sitzt. Im Sterben feuert er nochmal. Wahrscheinlich *Auto Answer*. Egal. Endlich reicht das Geld für *Quick*. Und weiter geht's mit doppelter Geschwindigkeit.

Midi-Maze II ist ein Action-Spiel erster Sahne. Der Vorgänger ist schon fast ein Klassiker, denn er begeistert schon seit Jahren viele Atari ST-Besitzer. Bis zu 16 Rechner können, über Midi verbunden, an einem Spiel teilnehmen. Jeder der Spieler steuert eine grinsende Kugel durch ein Labyrinth, das er in 3D-Sicht auf dem Bildschirm hat. Das Ziel des Spiels ist so einfach wie martialisch. Einzeln oder in Teams jagt man sich durch das Labyrinth und versucht alles, was sich bewegt, umzunieten.

Doch sollte man sich nicht gleich durch die Einfachheit oder den kriegerischen Aspekt des Spiels abschrecken lassen. Die Faszination liegt darin, daß die grinsenden Kugeln aus Bits nicht von einem hirnlosen Algorithmus gelenkt werden, sondern von Menschen, die sich immer wieder neue Strategien und Tricks ausdenken. In einer Gruppe ab vier Spielern macht es soviel Spaß, daß man leicht Stunden vor dem Bildschirm verbringt. Wen die Schießerei stört, der kann sich ja vorstellen, daß man Schwarzer Peter spielt, und wem zuerst drei schwarze Peter zugeworfen wurden, der hat verloren.

Jeder der in der ersten Version eine größere Vielfalt an Aktionsmöglichkeiten vermißte, der dürfte jetzt getröstet sein. Mit dem Geld, das man durchs An- und Abschießen erhält, kann man in Midi-Maze II viele nette Überraschungen kaufen. Das fängt bei Übersichtskarten des Labyrinths an, auf denen für etwas mehr Geld auch die Gegner eingezeichnet sind, und endet mit kompletter Unsichtbarkeit und der Fähigkeit, durch Wände zu laufen. Je nützlicher eine Fähigkeit ist, desto mehr muß man dafür bezahlen. Und da man mit einer vorher festgelegten Menge Geld gewinnt, sollte man es sich gut überlegen, ob und für was man das Geld ausgibt.

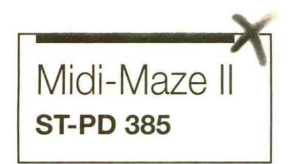
Da gibt es Sachen wie *Got You* und *No Got You*. Besitzt man ersteres, so wird man nicht verwundet, sobald man getroffen ist. Stattdessen bekommt der Angreifer die Auswirkungen seines eigenen Schusses zu spüren. Es sei denn, dieser besitzt *No Got You*, dann ist die Wirkung von *Got You* aufgehoben. Da *Got You* teurer als *No Got You* ist, ist es relativ leicht, sich zu schützen, und man muß sich überlegen, ob man das Geld für *Got You* ausgeben will. Auf alle Fälle nützlich sind *Reflective Shot*, *Fast Shot*, *Quick* und *Auto Answer*. Ersteres läßt den Schuß an den Wänden des Labyrinths abprallen und eignet sich deshalb hervorragend zum um die Ecke schießen. *Fast Shot* verleiht dem eigenen Schuß doppelte Geschwindigkeit, und *Quick* verdoppelt die Bewegungsgeschwindigkeit der Spielfigur. *Auto Answer* schießt, sobald man getroffen wird, in die Richtung zurück, aus der der Angriff kam. Das ist besonders für Leute, die einem in den Rücken schießen, eine böse Überraschung. Da die Geldmenge, die zum Gewinnen nötig ist, sowie alle Zusatzeigenschaften, die man erwerben kann, für jeden Spieler einzeln festgelegt werden, ist auch ein Spiel in einer Gruppe aus Midi-Maze-Kennern und Anfängern interessant.



Mnsich leicht Überblick über ein Labyrinth verschaffen.

Midi-Maze II kann man mit Maus oder Joystick, in Schwarzweiß oder Farbe und auch in selbstgebauten Labyrinthen spielen. Es ist schneller als der Vorgänger und es können bis zu 27 verschiedene Extras aktiviert werden. Das Spiel ist Shareware und wer sich registrieren läßt, bekommt ein Midi-Maze-T-Shirt. Also auf zur nächsten Midi-Maze-Session!

Manuel Chakravarty





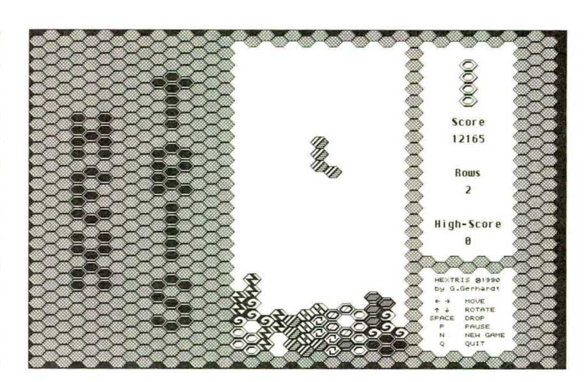
Sechseck-Tetris

Das genial-einfache Spielprinzip von *Tetris* scheint in letzter Zeit immer mehr Programmierer zu immer neuen Varianten dieses Genres zu inspirieren. Nach *Columns* (ST-PD 273), bei dem „Farbmuster“ richtig kombiniert werden müssen, *Setz_ab* mit einer dritten Dimension (SP-PD 346) oder *Leblon* (ST-PD 374), bei dem auf mehreren Feldern gespielt wird, findet sich auf der Diskette Nummer 374 nun mit *Hextris* eine weitere Version.

Im wesentlichen ist dieses Programm von G. Gerhardt ein „normales“ *Tetris*, das nun aber auf Sechsecken basiert. Die herabfallenden Figuren sind dementsprechend gestaltet und verhalten sich beim Rotieren aufgrund der schrägen Flächen etwas ungewohnt. Die Steuerung erfolgt über die Pfeiltasten des Cursor-Blocks. Ferner erreicht man mit *P* eine Spielpause, mit *N* ein neues Spiel, und mit *Q* kann man schließlich das Programm verlassen.

Tetris-Fans, die die normale Version schon perfekt beherrschen und etwas Abwechslung suchen, sollten ruhig einmal *Hextris* ausprobieren. Es ist sehr ähnlich, aber doch anders, so daß es eine neue Herausforderung darstellt. Einziges Manko dieses Spiels ist, daß Score wie Highscore zwar angezeigt, aber nicht abgespeichert werden, und es bei Spielende (keine Figur läßt sich mehr setzen) einfach stehenbleibt und auf ein *N* wartet.

thl



Hextris
ST-PD 374

Maß halten

In Deutschland ist immer alles sehr schön geregelt: Seit 1868 gibt es den Meter und davon abgeleitet Kilometer (das tausendfache) sowie Millimeter (ein tausendstel). Flächen- (Quadratmeter) und Raummaße (Kubikmeter) funktionieren entsprechend. Alle Angaben lassen sich prima durch einfaches Verschieben des Dezimalkommas ineinander umrechnen. Bei den Angelsachsen dagegen ist alles ein bißchen anders: dort mißt man in Inch (mit dem deutschen Wort Zoll ist wiederum eine andere Länge bezeichnet), Foot oder Yard. Auch von Unzen und Gallonen ist die Rede.

Wenn Sie nun einmal mit dem Problem konfrontiert werden, aus diesen etwas unhandlichen Maßangaben verständliche Zahlen zu machen, können Sie entweder im Lexikon nachschlagen und den Taschenrechner zücken oder eine Diskette ins Laufwerk schieben und das Umrechnungsprogramm von Udo Elger starten. Dieses in GFA-BASIC 2.0 geschriebene Programm (Quell-Code dabei) rechnet nach Bedarf die verschiedenen Angaben folgender Systeme entsprechend um: Gewicht, Leistung, Länge, Fläche, Druck, Rauminhalt, Temperatur und Arbeit. Das ist nicht nur besonders praktisch, wenn man sich mit amerikanischen Programmen oder Büchern befaßt, sondern auch bei den manchmal etwas befremdlichen Neubezeichnungen in der Technik: so wurde - zumindest offiziell - aus PS (Pferdestärken) kW (Kilowatt) oder aus der Kalorie die Einheit Joule. Auch hier rechnet das Programm nicht nur um, sondern zeigt zusätzlich noch eine kleine Umrechnungstabelle an.

thl

```
Einheiten-Umrechner

Länge                     12 Zoll

SI Einheit - Meter (m)
1 Meter  (m)  = 100 cm          = 304,8 mm
              = 1000 mm
              = 39,37 Zoll      = 30,48 cm
              = 3,28 ft
              = 1,094 yd
1 Yard   (yd) = 3 ft            = 0,3048 m
              = 36 Zoll
              = 0,9144 m
1 Fuß    (ft) = 12 Zoll         = 0,99999999999 ft
              = 0,3048 m
1 Zoll        = 0,0254 m
              = 2,54 cm         = 0,33333333333 yd
```

Masze
ST-PD 354



Adressbuch mit Komfort

Es ist ein altes Problem: Will man seine Adressen oder andere Daten elektronisch verwalten, stehen einem Dutzende von mehr oder weniger brauchbaren Datenbanken zur Verfügung. Mit Ihnen kann man Anschriften und Geburtstage von 10.000 Leuten speichern, sortieren und in Serienbriefen ausdrucken. Doch in der Praxis des privaten Anwenders liegen die Anforderungen oft etwas anders: Wichtige Adressen kennt man auswendig, und darüber hinaus benötigt man nur eine Handvoll weiterer Anschriften für individuelle Briefe.

Was liegt also näher, als dafür eine Miniaturdatenbank als Accessory zu konzipieren? Das Programm *Addresshelp* von Andreas Jäckel bietet die Möglichkeit, maximal 75 Anschriften zu verwalten und über die Menüleiste jederzeit abrufen zu können. Arbeitet man außerdem mit der Textverarbeitung *1st_Word* oder *That's Write*, lassen sich die Texte direkt in das aktuelle Dokument übertragen! Dabei lassen sich auch zusätzliche Leerzeilen sowie ein Anredetext einfügen. Das spart nicht nur Tipparbeit, sondern hilft auch, Schreibfehler zu vermeiden.

Das Accessory dürfte für den privaten Gebrauch eine gute Hilfe sein, wobei es nicht funktionsüberladen ist. Schade ist aber, daß das direkte Einfügen in den Text tatsächlich nur mit *1ST_Word* oder *That's Write* funktioniert. Alle anderen Programme, die ich daraufhin getestet habe, hatten nur ein müdes Cursor-Zucken übrig oder schrieben alles in eine Zeile, wenn Sie überhaupt Accessories zuließen. Ein schönes Detail ist auch, daß sich für die Adressen ein separates Suchwort eingeben läßt, daß nichts mit dem Adreßtext zu tun haben muß.

Für alle, die feststellen mußten, daß dieses Programm nicht mit ihrer Textverarbeitung zusammenarbeitet, noch ein kleiner Tip: Mit den selbstdefinierbaren Makros einer Textverarbeitung kann man in der Regel nicht nur Arbeitsschritte und Floskeln, sondern auch Anschriften speichern und einfügen. Ferner lassen sich über Werkzeugkiste (ST-PD 356) oder Harlekin (MAXON) Tastenmakros definieren, die in allen Programmen aktiv sind. Allerdings besteht dann Kollisionsgefahr mit Tastaturkürzeln.

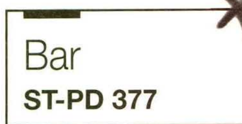
thl



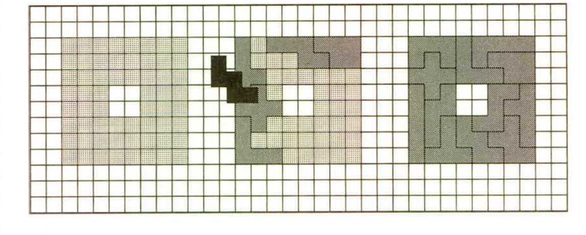
Feuchtfröhlich

Bar von Roland Fricke ist eine originelle Rezeptverwaltung für Cocktails. Das Programm zeichnet sich durch gute Bedienbarkeit aus und fällt durch einen lustig animierten Barkeeper aus dem Rahmen. In der ersten Version konzentrierte es sich noch auf alkoholische Getränke. Entsprechend dem gestiegenen Gesundheitsbewußtsein und der Verantwortung im Straßenverkehr sind in der Rezeptsammlung nun auch alkoholfreie Getränke zu finden, die mit einem Sternchen gekennzeichnet sind (*). Ferner wurde der Zeichensatz geändert, so daß nun auch ç, é, è, ú, ñ sowie $^1/_2$ und $^1/_4$ zur Verfügung stehen. Wie bisher wird beim Start des Programmes automatisch die Datei *STANDARD.-BAR* geladen. Neu ist allerdings, daß man nun auch beliebige andere Dateien laden oder im Speicher miteinander kombinieren kann. Doppelte Eintragungen werden anhand des Namens und der Zusammensetzung automatisch aussortiert. In einer Datei finden ca. 400 verschiedene „Giftmischungen" Platz. Dem Update beigefügt sind drei Sammlungen mit zusammen ca. 170 verschiedenen Getränken.

thl







Denk mal!

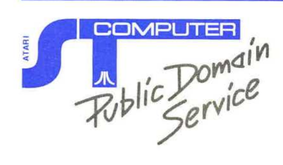
Auf den ersten Blick erinnert das *Pentomino* an das alte chinesische Puzzlespiel „Tangram", bei dem man aus verschiedenen Drei- und Vierecken schattenrißähnliche Bilder legen muß. Bei diesem Spiel bestehen die Figuren aus Kombinationen von jeweils fünf Quadraten, so daß es zwölf verschiedene Anordnungen gibt. Auch hier müssen in einem Raster vorgegebene Bilder aus den einzelnen Teilen zusammengesetzt werden. Dabei steht jedes Muster allerdings nur einmal zur Verfügung. Einzige Hilfe ist das Drehen der Figur, um sie in eine Lücke einzupassen. Bei dieser Umsetzung von G. Gerhard werden 38 solcher Rätsel mitgeliefert, die man ohne Zeitdruck und in beliebiger Reihenfolge lösen kann (ST-PD 374). Sollten einmal Zweifel an der Lösbarkeit aufkommen, kann man sich über den Menüpunkt *Lösung* die richtige Kombination anzeigen lassen. Ferner gibt es einen *Editor*, mit dem man die Aufgaben auf maximal 99 erweitern kann.

Pentomino ist kein typisches Computerspiel. Wer keinen Rechner besitzt, spielt es in der Regel kostengünstig mit selbst ausgeschnippelten Papier- oder Pappfiguren. So kann man es auch unterwegs ohne Laptop spielen. Die Computerversion kann einen natürlich auf den Geschmack bringen, sich zur Abwechslung einmal mit ähnlichen Spielen ohne „High Tech" zu beschäftigen (Literatur dazu ist nicht gerade selten).

thl



ST-COMPUTER PUBLIC DOMAIN



Liebe ST-Computer PD-Freunde,

auch in diesem Monat gibt es wieder eine große Zahl von Updates, mit denen wir wie immer die Disketten aktualisieren. Sie können daher sicher sein, in unserer Serie immer die aktuellsten Versionen zu finden, die uns die Autoren zuschicken, auch wenn sie mal nicht in der Update-Liste explizit erwähnt werden. Sollten Sie die Disketten von irgendeinem PD-Copyshop erhalten, achten Sie bitte auf die Versionsnummern, denn nur sehr wenige dieser Händler fragen bei uns nach den Updates. Nebenbei: es gibt keine offiziellen Zweigstellen oder Kopierservices oder ähnliches von der ST-Computer-PD-Serie, auch wenn einige sich als solche ausgeben.

Ihre ST-Computer-Redaktion

TOPAKTUELL IN DIESEM MONAT



SPRACHE

PD-Scheme ist eine experimentelle Programmiersprache. Sie umfaßt weitgehend den Scheme-Standardbefehlssatz. Genau wie Lisp, ist Scheme besonders dazu geeignet, symbolische Aufgaben zu bewältigen. Samt speziellem Editor und Source-Code in GFA-BASIC. (s/w)



DIVERSES

COMMAND: Äußerst leistungsfähiger Kommando-Prozessor (CLI). Bereits in früheren Versionen in der ST-PD Serie.

Trans ST 64: Datenübertragung von einem C64 auf den ST. So können alte Bilder (PICS), Zeichensätze (FONTS) und Texte (in Commodore-ASCII oder im Vizawrite-Format) auf dem ST weitergenutzt werden. Inklusive Bauanleitung für entsprechendes Verbindungskabel. [S, C64 erforderlich (soso).]

IBMASCII: Wandelt IBM-Textdateien in Atari ST-Format und zurück. (s/w)



DIVERSES

ASS_MAC: Bei den Quelltexten handelt es sich um eine Makrosammlung für den GFA- und andere Assembler, die die Aufrufe des Betriebssystems beträchtlich vereinfacht. Sie umfaßt AES-, VDI-, TOS-, und TOS30(TT)-Makros.

GTOOLSI: Graphic Tool Science. Programm zur grafischen Darstellung von Werten und zwar als Punkt-, Linien-, Balken- und 3D-Balkengrafik. Verarbeitet ASCII-Dateien oder Werte aus eigenem Editor. Darstellung von Standardabweichung. (s/w)

MULTITOOL: Accessory mit einigen praktischen Features. PIC-Anzeigen, Cursor und Maus ein- und ausschalten (praktisch, wenn irgendein Programm diese einfach ausschaltet), schnelle grafische Diskinfo über alle angeschlossenen Laufwerke, Telefonregister, Dateien kopieren/verschieben und Hardcopy auf Disk.

NECPRINT: kleines nützliches Programm zum Drucken von Hardcopies auf NEC Px-Druckern und kompatiblen. Ausschnitt, verschiedene Größen, light.

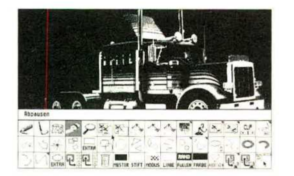
SHOW ME: Programm zum Anzeigen von Textdateien im Desktop. Texte können vor- und rückwärts geblättert werden. Benutzt Proportional-Font. Zeigt auch 1st_Word-Dateien vernünftig an.

SPEEDER: Verdoppelt die Lese- und Schreibgeschwindigkeit des Diskettenlaufwerks. Läuft auf allen RAM/ROM-TOS-Versionen (1.0-1.6).

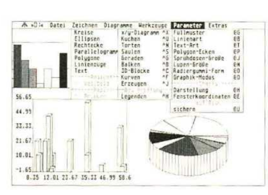
NOTEPAD: Einfaches, aber sehr praktisches Notizprogramm (ACC). Laden, Speichern, Drucken, Modem schicken. Als Accessory immer zu erreichen.



GRAFIK



ST DESIGNER 1.5: Zeichenprogramm mit hervorragenden Funktionen. Einige Besonderheiten: Pinsel (mit Füllmustern), Schwamm, Appausen, Verlauf, stufenloses Drehen, Zoomen, Clipboard, und Bézierkurven. Weiterhin zeichnet sich der Designer durch eigene Vektor-Fonts aus, die stufenlos vergrößert und gedreht werden können. Ein leistungsstarker Font-Editor ermöglicht die Kreation eigener Fonts. (S,s/w)



CHARTS_3: Programm zur Erstellung von Geschäftsgrafiken aller Art. Kuchen-, Säulen-, Linien-, Punktgrafik in 2D- und 3D. Ein universeller Dateneditor sorgt für leichte Eingabe. Zur weiteren Gestaltung der Grafiken sind einfache Grafikfunktionen eingebaut. (s/w)



ASSEMBLER

ASM63: Cross-Macro-Assembler für die Prozessortypen HITACHI 6301, MOTOROLA 6800, 6801, 6802 und ähnliche Prozessoren, sowie die darauf aufbauenden Microcontroller von HITACHI und MOTOROLA (z.B. HD 63701). Stark erweiterte Version (2.1).

ASMSHELL: Assembler-Shell ASMSHELL zum bequemen Arbeiten mit ST und einem Prozessor-Board. ASMSHELL arbeitet mit UNITERM, TEMPUS und JUNIORPROMMER problemlos zusammen.

ASMSHELL und ASM63 werden zur Ausbildung im Labor für Mikrocomputertechnik an der FH-Berlin der DBP-Telekom eingesetzt.



UTILITIES

DIVPATCH: Diverse Patches. Dient vor allem dem nachträglichen Anmelden einer Floppy im Desktop ohne Neustart, Step-Ratenumschaltung für 5 1/4"-Laufwerk. (s/w)

1.HILFE: Retten gelöschter Dateien und von Daten auf einer zerstörten Diskette. Sehr hilfreiches Programm. (s/w)



DESKHANDLER: Programm zum komfortablen Erzeugen einer DESKTOP.INFO. Sämtliche Einstellungen können in Menü ausgewählt werden, besonders das Anmelden von Applikationen (z.B. DOC, TXT, ASC auf WORDPLUS) wird unterstützt. (s/w)

DIS65: Disassembler für 6502-Code. Ideale Ergänzung zum 6502-Cross-Assembler von PD 358. Entwickelt wurde er übrigens mit Sozobon C (PD 240/241) und Gulám (PD 144).

FLAG 1_4: Programm zum Setzen des Fastload-Bits (schnelleres Starten eines Programms) und des Fast-Bit im TT, das angibt, ob ein Programm in das ST- oder in das schnelle 32-Bit-RAM geladen wird. (ab TOS 1.4)

HDREEE: gibt eine schnelle Gesamtübersicht über den Belegungsgrad der angeschlossenen Festplatte und deren Partitions aus.

NOTIZ: Notizblock als Accessory. Praktisch für Aufnahme wichtiger Ideen, Telefonnummern,...(s/w)

STARTUP: Boot-Hilfe für den AUTO-Ordner: Uhrzeit resetfest einstellen, Accessories und AUTO-Ordner-Programme wählen. Schnell. Sicher. UNDO möglich. V 3.3.

RAMFREI: Zeigt den vorhandenen, freien und belegten RAM in Ihrem ST in Bytes und in Prozent an. Führt auf Wunsch einen Warm- oder Kaltstart durch. Kann den Systemspeicher auf Wunsch verkleinern (praktisch zum Testen neuer Programme). Auch für TT-Speicher. V 3.1 (s/w)

KOMMANDO: Kommandozeileninterpreter. Eignet sich prima für Shells (z.B. Gemini). Durch Schieben einer Batch-Datei auf das Kommando-Icon wird diese ausgeführt. (S)

MACTEXT: konvertiert Zeichen und Zeilen-Enden von Mac-Texten für Wordplus, o.ä.

MEGAROY2: Speichermonitor mit anspruchsvollen Befehlen, z.B. grafische Darstellung des RAM-/ROM-Speichers, Suchen von Text und Icons, Ausschneiden von Speicherbereichen (Icons) als Datei. (s/w)



UTILITIES

PYROSAVE: Bildschirmschoner mit Feuerwerk. Einstellbare Zeit. (s/w)



UHR: Tolle Analoguhr als ACC und PRG. Weckfunktion, Westminster-gong, Zeitverstellung per Maus.

DIGIUHR: Blendet eine Digitaluhr in der Menüleiste ein. (s/w)

SYSTEM: Gibt intere Infos über den Rechner. Modell, RAM-Bestückung, angeschlossene Floppy-Laufwerke, BLITTER, TOS-Version, TOS-Datum, GEMDOS-Version, GDOS, VT52-Emulator, Gesamtspeicher (MMU+phystop), freier Speicher, Step-Rate, Verify und GEM-Fonts. (s/w)

SHOW_ME: Programm zum Anzeigen von Dateien im Desktop, jedoch mit Funktionen zum Vorwärts- und Rückwärtsblättern.

ph_res2: Schnelleres Booten durch Verhinderung des Löschens des Restspeichers bei Auto-Ordner Programmen und ACCs.

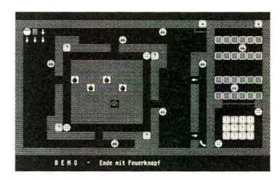
VOLUME: Dient zum Ändern des Diskettennamens.

DUMPVECT: Gibt die Belegung der ST-Systemvektoren auf dem Drucker aus.

FILESAFE: File-Verwaltung, die Festplatten- und Disk-Laufwerke unterstützt. Dateien können hiermit verwaltet werden, in verschiedene Kategorien aufgeteilt, gesucht, gedruckt und vielseitig bearbeitet werden. (s/w)



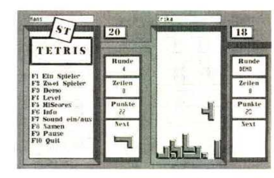
SPIELE



EISKALT: Actionspiel, bei dem Überblick und Ideenreichtum gefragt sind. In vielen Levels lauern immer neue Hindernisse und Gefahren. (Joystick,s/w)

NEWTRIS: Kombination von TETRIS und COLUMNS. 2 Varianten, die viele Stunden Spielfreude garantieren.

SH: Superhirn mit mehreren Spielvarianten (Doppelte erlaubt, 3 Level,...)

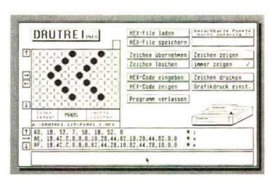


STTETRIS: fantastische Tetris-Variante für 2 Spieler. Jetzt kann man parallel zu zweit dem süchtig machenden Hobby frönen. (s/w)

MANAGERTEST: Beurteilt nach vielen Fragen, ob Sie ein Managertyp sind oder nicht. (s/w)



DRUCK-UTILITIES

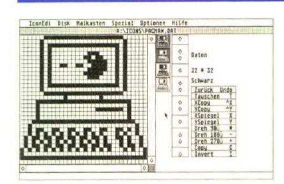


DRUTREIB: Programm zum Erstellen von Wordplus-Druckertreibern. Sonderzeichen können bequem per Maus gezeichnet werden. Drutreib generiert daraus automatisch eine HEX-Datei, die dieses Zeichen später als Grafikzeichen druckt. Vorbei sind die aufwendigen Definitionen per Karoblatt und Taschenrechner. (s/w)

GGINST: Programm zur Umwandlung von HEX- in CFG-Files. Speziell für Druckprogramm aus ST-Computer 4/90.

SOKRATES: Programm, das es dem Anwender erlaubt, griechische Texte einzugeben. Unterstützt Eingabe aller griechischen Zeichen, dazu Spirititi (Asper und Lenis), Zirkumflex und Akut, jeweils in allen Kombinationen mit den Vokalen. (S)

ST-COMPUTER PUBLIC DOMAIN



IconEdi: Editor für Icons. Viele nützliche Befehle, wie etwa Drehen und Verschieben. Ausgabe als Sprite, Maus und RSC-Datei. Leicht eingeschränkte Version, aber gut funktionsfähig. (s/w)

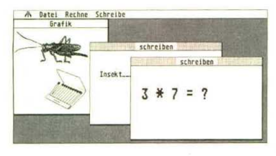
HEINZEL: Accessories zum Senden von Steuercodes an Drucker. Zusätzlich ist ein schneller Drucker-Spooler eingebaut. Spezialversionen für NEC- und Epson-LQ-Drukker.



LERNEN



PD_CMAMP: Vokabellernprogramm mit 800 englischen und über 2000 italienischen Worten. Umfangreiche Auswertungen, Erweiterbarkeit, Mehrfachbedeutungen u.v.m. machen dieses Programm zu einer großen Lernhilfe. (s/w)



FÖRDER: Das Programm dient zur Vertiefung von Rechen - und Rechtschreibfertigkeiten. Verschiedene Schwierigkeitsstufen. Im Schreibteil sind zusätzlich Bilder vorhanden, deren Bedeutung man eingeben muß, dadurch höhere Motivation. Mit eigenen Bildern erweiterbar. Ergebnis wird festgehalten, dadurch Steigerungsmöglichkeit. (s/w)





389 bis 399 Paket TeX

Achtung! Komplettes TeX-Paket, bestehend aus dem kompletten TeX 2.0, Metafont, Fonts **plus** TeXDraw, ZPCAD.
alle 11 Disketten für DM 89.-

MACHEN SIE MIT!

Möchten Sie ein selbstgeschriebenes Programm in unsere PD-Sammlung geben, um es auch anderen Usern zugänglich zu machen? Kein Problem. Schicken Sie es uns auf einer Diskette zu, samt einer Bestätigung, daß es von Ihnen geschrieben wurde und frei von Rechten Dritter ist. Bei Fragen steht Ihnen die Redaktion gerne zur Verfügung.

MAXON Computer
ST-Computer PD
Industriestr. 26
W-6236 Eschborn

UPDATES

Folgende Programme wurden von den Autoren überarbeitet bzw. erweitert. Daher sind diese Versionen ab sofort auf unseren original PD-Disketten enthalten.

PD 32 AGoFinanzmathematik 1.4: verbesserte und überarbeitete Version
PD 142 SCANNER 1.6: schnellere und erheblich verbesserte Version
PD 144 TEXSHELL 2.1: Es ist eine Fehlerbereinigung vorgenommen worden.
PD 215 KLIMA 2.1: Mit Hilfefunktion. Es werden jetzt 204 Datensätze mitgeliefert.
PD 243 UNI_LIGA 1.56: Die Tabellenausgabe wurde verbessert und ein Accessory-Aufruf ist nun möglich.
PD 280 FOTO 1.03: überarbeitete Version
PD 289 SKOOTER 1.1: Fehler wurden eliminiert
PD 305 Kartei 2.01: Das Programm arbeitet nun vollständig unter GEM.
PD 311 BUDGET 2.0: Diese Version ist völlig überarbeitet und liegt nun in erweitertem Umfang und voller GEM-Bedienung vor.
PD 318 TELDAT 2.31: Mausunterstützung ist erweitert worden
PD 318 1stSOUND 1.28: verbesserte Version mit erweiterter Suchroutine
PD 329 MORSEMASTER 2.0: überarbeitete und erweiterte Version
PD 338 TOPSHELL: Die Programme können nun auch über die Tastatur gestartet werden.
PD 341 Hüpfer 3.4: Abspeicherfehler wurde behoben und diese Version durch trigonometrische Funktionen erweitert
PD 342 HPGL-VDI-TREIBER 1.4: überarbeitete Version
PD 343 ZPrint 1.7: läuft auf allen ST- und TT-Auflösungen und die Formatieroptionen können wahlweise eingestellt werden
PD 362 EXPLODE 3.0: Es kann nun gegen einen Computer-Gegner gespielt werden.
PD 367 DirList 1.4: läuft nun auch in der mittleren Auflösung und auf allen TTs mit mindestens 640x400 Pixeln, kleinere Fehler wurden beseitigt.
PD 367 Readboot 1.7: Fehler wurden behoben und die Bedienung mit der Maus ist nun möglich.
PD 373 LORD 1.5: Alle Dialoge sind nun voll GEM-unterstützt

ABKÜRZUNGEN

1MB = mind. 1MB Speicher notwendig

s/w = nur Monochrom;
f = nur Farbe
S = Shareware

Schon gesehen?

"PD ROYALE - Das Beste aus der ST Computer Public Domain Serie" entweder direkt bei MAXON Computer für DM 34,- inkl. Versandkosten oder über den Buchhandel (unverbindlicher Verkaufspreis DM 29,-).

ISBN 3-927065-07-2

Die in PD ROYALE beschriebenen Programme sind auf über 60 Disketten verteilt. Wir haben uns daher dazu entschlossen, diese Programme neu gesammelt zusammenzustellen. Herausgekommen sind drei *'Pakete Royale'*, bestehend aus je 5 randvollen Disketten, gefüllt mit Programmen aus diesem Buch.

PAKET ROYALE A - Anwendungen
Minitext, Deluxe Fontmaster, Fontedit, Tiny Editor, Keyhelp, Brief, Last Word, Umlaut, Prn-Send, Speedwriter, ASCII-Edit, Printing Press, Little Painter, Showtime, Sticker, LQ 800, Hardcopy, Public Painter, WDR-Bild, SBASE, GEM-Calc.

5 Disketten DM 39,-



PAKET ROYALE B - Utilities+Unterhaltung
ARC, Sagrotan, Bitte ein Bit, Cruncher, FCOPY III, Hyperformat, Packer, ZOO, Calendar, Chooseboot, Deskedit, Dump, FSelect, Give Up, Goodies, James, JClock, Lock, Megamatic, Simple, ST-Klick, Termin, Uhr, Werkzeugkiste, X-Utility, CHR-Tree, Diskkatalog, Goodview, Hide, SuperFileCopy, XDir, XDirList, Bar, Biorythmus, ST-Kalender, Kalender, Shapes, Erdkugel.

5 Disketten DM 39,-



PAKET ROYALE C - Spiele
Tetrix, Stones, Columns, Diamond Miner, Solitaire, U-Boot, Laserschach, Superbreakout, Invaders, Mac Pan, Memory, Quiz, Wizzy, Go-Up, DGDB I+II

5 Disketten DM 39,-



DIREKT-VERSAND

Alle PD-Disketten unserer Sammlung gibt es nur direkt bei MAXON-Computer.

1. Schriftliche Bestellung

- Der Unkostenbeitrag für eine Diskette beträgt DM 10,-
- Hinzu kommen Versandkosten von DM 5,- (Ausland DM 10.-)
- Bezahlung per Scheck oder Nachnahme
- (Im Ausland nur Vorauskasse möglich)
- Bei Nachnahme zuzüglich DM 4,00 Nachnahmegebühr
- Ab 5 Disketten entfallen die Versandkosten (DM 5.- bzw. DM 10.-)
- Der Versand kann aus technischen Gründen **ausschließlich** gegen Nachnahme oder Vorauskasse erfolgen (auch für Händler!).

2. Telefonische Bestellung

MAXON-Computer GmbH
'PD-Versand'
Tel.: 0 61 96 / 48 18 11
Fax: 0 61 96 / 4 18 85
Mo-Fr 9⁰⁰ - 13⁰⁰ und 14⁰⁰ -17⁰⁰ Uhr

- Lieferung erfolgt per Nachnahme

Adresse:

MAXON-Computer GmbH
'PD ST-Computer'
Schwalbacher Straße 52
W-6236 Eschborn

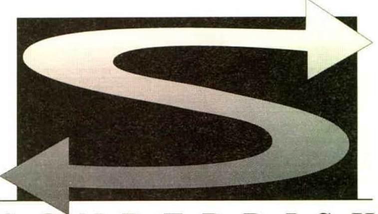
Nutzen Sie die PD-Karte in diesem Heft

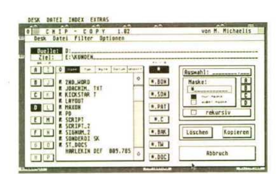
Immer up to date

	Programmname	Version	Daten	
	Adimens ST	3.1	N HM	
	Adiprog SPC Modula	1.1	N HM	
	Aditalk ST	3.0	N HM	
	Adress ST / Check ST	1.0	N H	
	Afusoft Morse-Tutor	2.0	N HML	
	Afusoft Radio-Writer	1.0	N HML	
	Afusoft Radiofax plus	1.0	N HML	1M
	AIDA	1.1	N HM	
	AnsiTerm	1.4	N	
	Arabesque	1.20	N H	
	Arabesque Profesional	2.00	N H	
	Assembler Tutorial	1.06	N HM	
	Banktransfer	1.0	N H	
	1st BASIC Tool	1.1	N HML	
	BTX-Börsenmanager	4.0	N H	
	BTX/VTX-Manager	3.0	N H	1M
	Calamus	1.09	N H	1M
	Cashflow	1.0	N H	1M
	Chips At Work	1.0	N HM	
	CIS-L&G	2.1	N H	2M
	CiSystem	2.1		2M
	Clix-Editor	2.15	N HM	1M
	Convector	1.01	N H	
	Creator	1.1	N H	
	Cubase	2.0		
	CW-Chart	8.0	N H	1M
	Daily Mail	1.2	N H	
	dBMAN	6.0	N HM	
	Diskus	2.0	N HM	
	dBMAN	5.10	N HML	
	Easybase	1.1	N HM	
	Easytizer	1.0	N HM	
	Easy Rider Assembler	2.04	N HM	
	Easy Rider Reassembler	2.31	N HM	
	Edison	1.00	N HM	
	fibuMAN	4.0	N H	
	fibuSTAT	2.3	N H	
	Flexdisk	1.4	N HML	
	FM-Meßtechnik	1.0.b	N HM	
	FTL Modula-2	1.18	N HM	
	Gadget	1.2.5b	N H	
	GEMinterface ST	1.1.	N HML	
	GFA-Artist	1.0	N L	
	GFA-Assembler	1.5	N HML	
	GFA-BASIC 68881	1.3	N HML	
	GFA-BASIC-Compiler	3.5	N HML	
	GFA-BASIC-Interpreter	3.5	N HML	
	GFA-Draft plus	3.01	N	
	GFA-Farb-Konverter	1.2	N H	
	GFA-Monochrom-Konverter	1.2	N ML	
	GFA-Objekt	1.2	N HM	
	GFA-Starter	2.0	N HML	
	GFA-Vektor	1.0	N	
	G+Plus	1.4	N HML	
	GrafStar	1.0	N H	
	Hänisch Modula-2	3.111	N HML	
	H.Modula-2-Runtime-Debugger	1.02	N HML	
	H.Modula-2-ONYX-Assembler	1.62	N HML	
	H.Modula-2-Window-Library	4.0	N HML	
	H.Modula-2-GEMplus-Library	2.0	N HML	
	Hard Disk Accelerator	1.0	N HML	
	Hard Disk Sentry	1.10		
	Hard Disk Toolkit	2.0	N HM	
	Harddisk Utility	3.0	N HM	
☛	Harlekin	2.0	N H	1M
	Imagic	1.1	N HML	
	Intelligent Spooler	1.10	N HML	
	Interlink ST	1.89	N HM	
	ISI-Interpreter	1.20	N HM	
	Junior Prommer	2.33	N HM	
	K-Resource	2.0	N HM	
	Kleisterscheibe	2.30	N HM	
	Label ST	1.0	N HML	
	Laser C (Megamax)	2.1	N HML	
	1st_Lektor	1.2	N HM	
	Lern ST	1.22	N HML	
	Link_it GFA	1.1	N HML	
	Link_it Omikron	2.0	N HML	
	MagicBox ST	7.78	N HM	1M
	Mathlib	3.0	N HM	
	Mega Paint II	2.30	N H	1M
	Mega Paint II Professional	2.31	N H	1M
	Megamax Modula 2	3.5	N HM	
	MGE Grafikkarte	1.27	N	
	MGP GAL-Prommer	2.0	N H	
	Micro C-Shell	2.70	N HM	
	MPe II plus	1.02	N H	1M

	Programmname	Version	Daten	
	Mr Print	3.0	N H	
	MT C-Shell	1.2	N HM	1M
	Multidesk	1.82	N HML	
☛	MultiGEM (PAM's)	1.0	N HML	
	Musix32	1.01	J H	
	NeoDesk	3.0	N HML	
	Notator	3.0		
	NVDI	1.0	N HML	
	Omikron Assembler	1.86	N HML	
	Omikron BASIC-Compiler	3.06	N HML	
	Omikron BASIC 68881-Compiler	3.06	N HML	
	Omikron BASIC Interpreter	3.03	N HML	
	Omikron DRAW! 3.0	3.01	N HML	
	Omikron EasyGEM-Lib	1.0	N HML	
	Omikron Maskeneditor	1.0	N HML	
	Omikron Midi-Lib	2.1	N HML	
	Omikron Numerik-Lib	1.2	N HML	
	Omikron Statistik-Lib	1.5	N HML	
	PAM's TERM/4014	3.012e	N H	
	PAM's TurboDisk	1.7	N HML	
	PAM's NET	1.1	N HML	
	PCB-layout	1.19	N H	
	PegaDress	1.0	N H	
	PegaFakt	2.0	N H	
	PegaStic	1.1	N H	
☛	Phoenix	1.0	N HML	
	phs-BTX-Box	6.1	N HML	1M
	phs-ST-Box	1.2	N HM	
	phs-Boxtalk	1.0	N HM	1M
	phs-Boxedi	1.0	N HML	1M
	Platon	1.45	N H	
	1st Proportional	3.13	N HM	
	Prospero Pascal	2.153	N HML	
	Prospero Fortran	2.153	N HML	
	Prospero C-Compiler	1.144	N HML	
	Prospero Developers Toolkit	1.111	N HML	
	Protos	1.1	N H	1M
	Querdruck2	2.05	N HM	
	Quick_Dialog	1.0	N HM	
	ReProk	1.10	N H	1M
	Revolver	1.1	N HML	1M
	Rufus	1.04	N HML	1M
	Scarabus	2.0	N H	
	Scigraph	2.0	J HM	
	Script	2.0	N HM	
	Search!	2.0	N HM	
	Signum! zwei	2.01	N H	
	Simula	2.1	N HML	1M
	Skylink	1.5	N H	1M
	Skyplot+	4.3	N H	1M
	Soundmachine II	1.0	N HM	
	SoundMerlin	1.01	N HM	
	SPC-Modula-2	2.0	N HML	
	Spectre 128	1.9	J HM	
	1st_Speeder 2	1.0	N HML	1M
	SPS ST	1.5	N H	1M
	STAD	1.3+	N H	
	Steuer-Tax 2.9	3.01	N HM	
	Steuer-Tax 3.9	3.01	N HM	
☛	Steve	3.0	N H	
	STop	1.1	N HM	
	ST Pascal plus	2.08	N HM	
	Supercharger	1.4	J H	
	Technobox Drafter/2	2.0	J H	1M
	Technobox CAD/2-ST/TT	1.4	J H	2M
	Tempus Editor	2.10	N HM	
	Tempus Word	1.0	N H	1M
	That's Write	1.51	N HM	
	Theca Librarian	1.0	N HM	
	Themadat	4.10		
	TIM	1.2	N H	
	TIM II	1.0	N H	1M
	Transfile ST 1600	1.1	N HM	
	Transfile ST 850	1.2	N HM	
	Transfile ST plus	3.1	N HM	
	Transfile ST E500	2.0	N HM	
	Transfile ST SF	2.0	N HM	
	Transfile ST IQ	1.4D	N HM	
	Turbo C	2.0	N HM	
	Turbo ST	1.8	N HML	
	UIS II + Hermes	2.5		
	V_Manager	3.1	N H	
	VSH Manager	1.0	N HML	1M
	WERCS Resource-Editor	1.0	N HM	
	Wordperfect	4.1	N H	
	Writer ST	2.0	N HM	
	Wordplus	3.15	N HML	

Irrtum vorbehalten! Daten-Legende : N = kein Kopierschutz, J = Kopierschutz, H = hohe Auflösung, M = mittlere Auflösung, L = niedrige Auflösung, 1M = mindestens 1 Megabyte, ☛ = Änderung gegenüber letzter Ausgabe



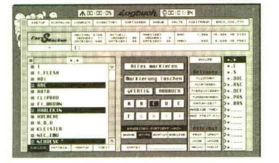




CHIPCOPY

Extrem flexibles Datei-Utility als Programm und Accessory. Mit Chipcopy kann man jederzeit nach zahlreichen Auswahlkriterien Dateien und komplette Ordner kopieren, verschieben oder löschen.
Durch logische Verknüpfung mehrerer Filter kann man optimale Auswahlkriterien schaffen und somit z.B. schnell Sicherheitskopien aller Texte der letzten Woche und zugehörendem Textprogramm anfertigen. Jeder Filter besteht aus mehreren Wildcards und kann zusätzlich auf Erstellungszeitraum begrenzt werden.
CHIPCOPY kann komplette Disketten kopieren und formatieren und weiterhin nach den selbigen Auswahlkriterien Verzeichnisse drucken.

CHIPCOPY [1]
SD 53 *DM 25.-*



FastSectorBackup 4.0

FastSectorBackup ist das ideale Tool für Ihre Datensicherung. Zum einen bietet es ein Image-Backup, welches komplette Partitionen sichert, und zum anderen ein sehr flexibles FileBackup. Damit lassen sich einzelne Dateien, welche nach Wildcards, Datum, Archiv-Bit oder einfach per Mausklick markiert werden, sichern. Weiterhin bietet FastSectorBackup die Möglichkeit, mehrere Backup-Vorgänge mit verschiedenen Markierungsarten in Batch-Dateien festzuhalten. Diese können dann automatisch ablaufen.

FastSectorBackup [1]
SD 35 *DM 25,-*

ORDNE HDB

Nach häufigem Schreiben und Löschen auf Festplatte sind die zusammengehörenden Teile einer Datei (Cluster) oft weit verstreut, was zu erheblichen Zeitverlusten führt. Das Programm ordnet die Struktur völlig neu, so daß alle Cluster einer Datei unmittelbar beieinander liegen. Der Plattenzugriff wird dadurch schneller.
Weitere Funktionen: Retten bzw. Regenerieren gelöschter Dateien, Umstrukturierung der Directory-Einträge, FAT-Analyse, Belegen defekter Sektoren, Ordner-Struktur zeigen, Namen (Platte/Ordner) ändern und anderes.
ORDNE HDB unterstützt die Treiber AHDI, CBHD, ICD, Eickmann und Vortex.

ORDNE HBD
SD 51 *DM 20.-*

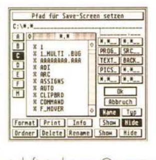




HELP!

Multi-Accessory

HELP! besteht aus vielen nützlichen Elementen, die als Accessory in GEM-Programmen bereitstehen: Kontrollfeld, Druckeranpassung, Druckereinstellung, Datei kopieren, Editor, Notizblock, Fileselector, Harddiskpark u.a.
HELP! verfügt über eine erweiterte Hardcopy in verschiedenen Größen und wahlweise mit Bildausschnitt (Graustufenkonvertierung bei Farbe). Ebenso kann der Bildschirminhalt auf Disk in gängigen Grafikformaten abgelegt werden.



Der HELP!-Fileselector bindet sich mit Optionen zum Drucken, Formatieren, Löschen und Umbenennen ins System ein und besticht u.a. mit frei definierbaren Gruppen, die sich aus mehreren Extentions zusammensetzen können (z.B. DOC, TXT, ASC, read*.*).
Ein Formelinterpreter, auch Taschenrechner genannt, ermöglicht die Berechnung komplexer Formeln in binär, octal, hex und dezimal, verfügt über Variablen, logische Verknüpfungen und viele mathematische und trigonometrische Funktionen.
Ein kleiner GEM-Editor in eigenem Fenster hilft beim schnellen Anzeigen oder Ändern von Texten. Sämtliche Funktionen können auch über eine Kommando-Shell aufgerufen werden.
Eine praktische Hilfe für viele Zwecke.

HELP!
SD 54 *DM 25.-*

KOALA

Der Monochromemulator

KOALA, der Monochromemulator ermöglicht es, Software, die für Monochrommonitor geschrieben wurde (z.B. SIGNUM!), auch auf einem Farbbildschirm laufen zu lassen.

Freie Einstellung der Bildwiederholfrequenz. So kann man zwischen hoher Bildrate oder hoher Rechenleistung wählen • Bildaufbau während Diskettenzugriff abschaltbar • Bildschirm-Hardcopy auf Disk (Farb- und s/w-Bild).

KOALA ist kompakt, schnell und für alle ST-TOS-Versionen (1.0-1.6).

KOALA [2]
SD 43 *DM 15.-*



BBAUM

BBAUM ist ein äußerst leistungsstarkes Tool für die Programmdokumentation von C-, PASCAL- und GFA-BASIC-Programmen. Vor allem die Einarbeitung in fremde Quelltexte wird vereinfacht, indem grafisch in Form eines Baumes die Funktions- bzw. Prozedurabhängigkeiten dargestellt werden.

BBAUM untersucht: C-Quelltexte • PASCAL-Quelltexte • GFA-BASIC-Quelltexte (2.0, 3.0 und 3.5) • DMP-Dateien (interne Baumstruktur) • Verzeichnisse (Struktur Ihrer Festplatte/Diskette)

BBaum verwaltet Includes bzw. ausgelagerte Programmteile und fügt sie automatisch an die entsprechenden Stellen im Hauptprogramm an. Wahlweise werden auch die Routinen dargestellt, die in der System-Library definiert sind (z.B. *printf* oder *getchar*).

BBAUM
SD 50 *DM 25.-*

FORMULA

2D-/ 3D-Plotter

Für mathematisch-wissenschaftliche Anwendung. Der eingebaute Formel-Interpreter beherrscht neben allen gängigen Operationen auch die Definition verschiedener Formeln in bestimmten Teilbereichen, logische Operationen und IF..THEN..ELSE. 3D-Grafiken lassen sich aus verschiedenen Blickrichtungen anzeigen und mit Schattierungen versehen.

FORMULA
SD 23 *DM 20.-*

HARDCOPY II

Die erste Farb-Hardcopy für den ST

Universelles Hardcopy-Tool. S/W- und **Farb**-Hardcopy auf *allen* Druckern in *allen* Größen, Screendump auf Disk, Formatkonvertierung, läuft als Accessory, einfachste Bedienung, optimale Druckqualität.

HARDCOPY II
SD 15 *DM 15.-*

PANDA

Der Farbemulator

Der Farbemulator simuliert die Farbauflösungen des ST auf einem monochromen Monitor (SM,124,..). Dadurch kann man auch Farbprogramme laufen lassen, die sonst einen zweiten Monitor erfordern.

PANDA [1]
SD 18 *DM 15,-*



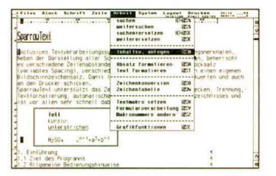
1stTrenn

vollautomatische Silbentrennung für 1stWordPlus

Darauf haben viele schon lange gewartet. Eine schnelle, automatische und präzise Silbentrennung für 1stWordPlus. 1stTrenn ersetzt die eingebaute Trennhilfe völlig, d.h. wird automatisch anstelle der eingebauten manuellen Trennung aktiviert (F10).

- arbeitet im Hintergrund (Accessory), 1stWordPlus muß nicht verlassen werden
- schnelle Trennung
- wahlweise mit Bestätigung oder vollautomatisch.
- hohe Trefferquote von über 98%, d.h. etwa eine falsche Trennung bei 8 Seiten Text.
- zusätzliche Autosave-Funktion des aktiven Textes
- läuft auf den deutschsprachigen 1stWordPlus Versionen 1.89, 2.02 und 3.15

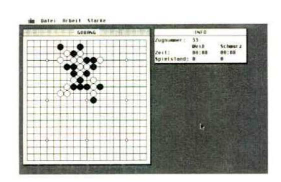
1stTrenn
SD 42 *DM 25.-*



SparrowText

Exklusives Textverarbeitungssystem mit besonderen Leistungsmerkmalen. Neben der Darstellung aller Schriftarten auf dem Bildschirm beherrscht es verschiedene Zeilenabstände, Proportionalschrift im Blocksatz (variables Spacing), verschiedene Font-Größen und vor allem einen eigenen Bildschirmzeichensatz. Damit lassen sich Sonderzeichen entwerfen und auch an den Drucker schicken.
SparrowText unterstützt das Zeichnen von Linien und Rechtecken, Trennung, Textformatierung, automatische Erzeugung eines Inhaltsverzeichnisses und ist vor allem sehr schnell dabei.
Als besonderen Leckerbissen ermöglicht es Formularverarbeitung, die sich hervorragend zum Ausfüllen von Briefbögen, Adreßfeldern oder allgemeinen Formularen eignet. Die Eingabefelder lassen nach Wunsch auch Eingabebeschränkungen (z.B. nur Zahlen) zu und bieten daher die Möglichkeit, gewisse Felder miteinander aufzuaddieren. Weiterhin kann man diese Felder automatisch ausfüllen lassen, da SparrowText Daten von einer Datenbank importieren kann und diese in die Felder einträgt. Dadurch läßt sich das Programm für Serienbriefe, Zeugnisse oder gar Rechnungen/Mahnungen einsetzen.

SparrowText [1]
SD 37 *DM 25,-*

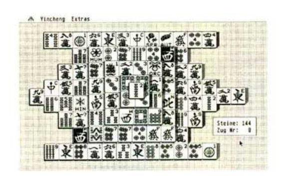


GOBANG

Ein Strategiespiel

GOBANG ist ein klassisches Brettspiel, bei dem abwechselnd Steine auf das Spielfeld gesetzt werden, wobei es gilt, 5 Steine in einer Reihe (senkrecht, waagrecht oder diagonal) zu plazieren. Der Computer bietet hier einen spielstarken Gegner, der nicht so leicht zu besiegen ist.
Neben dem Laden und Speichern einer Partie verfügt Gobang über verschiedene Spielstärken; vom Anfänger bis zum Profi. Auch die Blitzpartie, bei der jeder Spieler nur 30 Sekunden Bedenkzeit pro Spiel hat, bietet ihren speziellen Reiz. Ist man in einer schwierigen Lage, hilft der Rechner gerne mit einen Zugvorschlag aus.

GOBANG
SD 49 DM 15.-



YINCHENG

Dieses Spiel beruht auf dem alten chinesischen Patience-Spiel Mah-Jongg. Es geht darum, das mit 144 Spielsteinen gefüllte Spielfeld zu entleeren, wobei immer nur zwei zueinander passende und nach bestimmten Regeln positionierte Steine entfernt werden dürfen. YINCHENG beinhaltet eine zwei- und eine dreidimensionale Spielvariante, die sich zwar in den Regeln, doch kaum in der Spielqualität unterscheiden.

YINCHENG
SD 45 DM 20.-

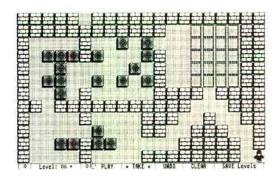


ODYSSEUS

Schachprogramm

Hinter Odysseus steckt ein spielstarkes und komfortables Programm. Die Züge lassen sich leicht per Maus eingeben. Es verfügt über eine Zeit- und eine Tiefensteuerung (bis zu 12 Halbzüge) und beherrscht den Turniermodus. Die beigefügte, jederzeit erweiterbare Bibliothek erlaubt dem Programm den Zugriff auf wichtige Züge. Mit ihm kann man Partien speichern, nachspielen und analysieren lassen.

Odysseus
SD 41 DM 25.-

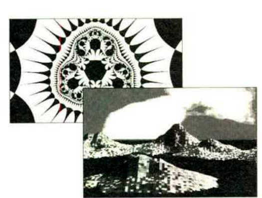


TAKE_1

Denkspiel

Die Regeln von Take_1 sind einfach - schieben Sie alle Diamanten auf die vorgesehenen Leerfelder. Leider gibt es ein paar Probleme... Der Vogel, den Sie mit den Cursortasten bewegen, kann immer nur einen Diamanten schieben und außerdem kann er sie nur schieben, nicht jedoch ziehen - das bedeutet, daß Diamanten an der Wand oder in einer Ecke nicht mehr verschoben werden können. Hier ist Taktik und Weitblick gefragt, denn so einfach, wie es im ersten Augenblick ausschaut ist es wirklich nicht. Durch den integrierten Level-Editor, lassen sich eigene Screens erzeugen.

TAKE_1
SD 46 DM 15.-



Art Of Fractals

Expedition ins Land der Fractale. A.O.F. beginnt bei Apfelmännchen (jedoch in 3D), behandelt Julia-Mengen, Iterationen aus der Planzen und Tierwelt und entführt Sie in dreidimensionale Landschaften. Steile verschneite Gebirgshänge im Mondschein oder eine Meereslandschaft an einem wolkigen Tag? Das Programm berechnet und stellt sie dar. A.O.F. erzeugt Fantasielebewesen und läßt mathematische Pflanzen gedeihen. Lassen Sie sich diese Reise nicht entgehen, noch sind Plätze frei.

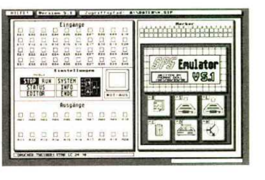
Art Of Fractals
SD 52 DM 20.-

Dialog Construction Set

für GFA-BASIC 3.x

Mit dem Dialog Construction Set (DCS) lassen sich auf einfache Art und Weise LST-Dateien erstellen, die den Programmcode zur Behandlung von Dialogboxen unter GFA-BASIC 3.0 enthalten. So ist es möglich, diese schnell und bequem in eigene Programme einzubauen. Als Voraussetzung wird natürlich weiterhin das Resource Construction Set (wird bei GFA-BASIC mitgeliefert) benötigt. Einfach mit dem RCS erstellen und dann mittels DCS den Programmcode generieren. Grundkenntnisse über Dialogboxen und GFA-BASIC-Programmierung sind aber weiterhin erforderlich.

DCS
SD 48 DM 15,-



SPS-Emulator V 5.1

für programmierbare Steuerungen

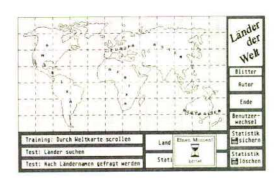
Unser SPS-Emulator baut auf einem SIEMENS PG 605-Programmiergerät in STEP 5 auf. Mit ihm lassen sich SPS-Programme schreiben, auf Simulationsbasis austesten, laden, speichern, ändern, ausdrucken und als FUP (Funktionsplan mit logischen Gattern) ausgeben. Enthalten sind ein Editor, ein Interpreter und FUP-Generator. Alle Befehle wurden voll im Siemens S5 Standart umgesetzt • 20 Timer als SE-, SA, SI-, SS-, SV-Timer zu verwenden • 20 Zähler (vorwärts/rückwärts), erhöhte Werte • Mehrfachzuweisungen nach einer Verknüpfung • wahlweise 20/40 Eingänge bzw. Merker • Schnellere Interpreterroutine (20-25%) • Startmerker für Autostart • Not-Aus-Merker/-Schalter • Blinkmerker: Vier Merker werden als astabiler Multivibrator angesteuert. • Sprungmarken (A-Z) • Komfortables Drucken der Awl • Die Merker-, Eingangs- und Ausgangsbezeichnungen können nun dezimal, hexadezimal oder byteweise bezeichnet werden. • Klammerbefehl —> U(,) • Oder vor Und —> O • Neuer Texteditor • Erhöhter Bedienungskomfort • Programmierung von Netzwerken

SPS Emulator V5.1
SD 14+ DM 25.-

DAME

Computerumsetzung des alten Brettspiels, wobei der ST einen spielstarken Gegner darstellt. Die Figuren werden per Maus angewählt, die Züge protokolliert und analysiert. Verschiedene Spielstärken, Zugvorschläge, Laden und Speichern einer Partie, sowie verschiedene Spielvarianten dürfen nicht fehlen.

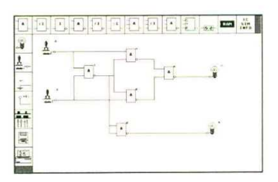
DAME
SD 29 DM 15.-



LÄNDER DER WELT

Geographie-Lernprogramm mit leicht verständlicher Bedienung. 'Länder der Welt' vermittelt die Lage der einzelnen Länder auf der Weltkarte. Wo liegt z.B. Togo? 'Länder der Welt' hilft weiter und sorgt mit seinen Trainings- und Prüfungsfunktionen dafür, daß der Anwender diese Frage nicht ein zweites Mal stellen muß.

Länder der Welt
SD 39 DM 15.-



ICSIM

Logik-Simulator

Das Programm simuliert das Verhalten von logischen Schaltungen. Bausteine und Verbindungen werden frei per Maus positioniert bzw. verbunden. Eine Schaltung läßt sich somit leicht austüfteln, testen und erst dann in die Praxis umsetzen.
Es sind die Logikbausteine nach DIN 40900 enthalten: AND, OR, NOT, NAND, NOR, XOR, RS-FF, KLEMME, LAMPE, SCHALTER, 0V und +5V.
Die Simulation wird als Impulsdiagramm oder Logiktabelle ausgegeben. Weiterhin liefert das Programm den Schaltplan und eine Liste der benötigten Bauteile.

ICSIM
SD 25 DM 20.-

DATEI LOGIK

Datenbank, die einfache Handhabung und große Flexibilität miteinander vereint. So ist es für jedermann möglich, sich ohne große Anstrengung eine Datenbank nach seinen Vorstellungen aufzubauen. Mit Hilfe des integrierten Formulareditors kann eine individuelle Abfragemaske erstellt, mit dem Etiketteneditor das Layout von Aufklebern oder Karteikarten für jeden Aufgabenbereich festgelegt und mit der Mailmerge-Funktion mit den Daten auch Serienbriefe erstellt werden.

Datei Logik
SD 36 DM 20.-



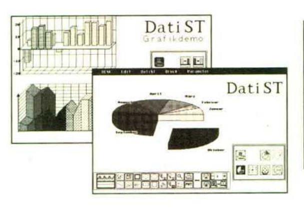
ASSOZIATIX

Assoziative Datenbank

Assoziatix ist eine assoziativ-Muster orientierte Datenverwaltung, die es ermöglicht aus einer großen Datenmenge bestimmte Gruppen auszufiltern und daraus dank schneller assoziativer Suche nach bestimmten Konstellationen, Zusammenhänge zu finden (z.B. Rasterfandung).
Mit Hilfe des Formulareditors können die Eingabemasken leicht am Bildschirm gestaltet werden, sogar mit Grafikeinbindung.

Einige Besonderheiten:
- Paßwortschutz, Export- und Importfunktion, Serienbriefe, Reportdokumentation Statistische Berechnung numerischer Werte Expertfunktion, Volltextsuche Grafikeditor: Spiegeln, Drehen, Zoomen. Balken- Linien und Kuchengrafik.

ASSOZIATIX (2 Disketten)
SD 27 a/b DM 30.-

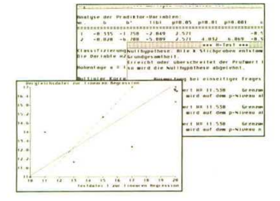


DATIST

Präsentationsgrafik

Grafiken sagen oft mehr als 1000 Zahlen, daher sollte man sich bei der Auswertung von Daten auf DatiST verlassen. DatiST stellt Ihre Daten als, Kuchen-, Reihen-, Balken-, Säulen- und Liniengrafiken dar, entweder in 2D oder 3D, gefüllt oder als Rahmen. Lage, Größe, Dehnung und der Nullpunkt einer Grafik lassen sich frei mit der Maus einstellen; dafür sorgen die iconisierten Pop-Up-Menüs. Im 3D-Modus kann gar die räumliche Perspektive frei variiert werden. Die so erzeugten Grafiken, lassen sich beschriften (z.B. mit SIGNUM!-Fonts) oder mit dem integrierten Zeichenprogramm bearbeiten, das vom Linienziehen über Blockoperationen bis hin zur Lupe alles bietet was man braucht. Um die Grafik zu Papier zu bringen bietet DatiST eine variable Druckeranpassung, die folgende Drucker unterstützt: Epson 9N/24N, NEC 24N, IBM PPR 24N, IBM AGM 24N, HP Laser, Atari-Laser!!.

DatiST [1]
SD 40 *DM 25.-*



STatiST

modulares Statistik-Programmpaket

STatiST ist ein umfangreiches Paket zur Auswertung statistischer Daten. Zu jedem Prüfverfahren werden sämtliche Ergebnisse mit den entsprechenden Wertungen und Kommentaren ausgegeben und, falls möglich, grafisch angezeigt. STatiST eignet sich für sämtliche, z.B. im Studium erforderlichen statistischen Auswertungen und macht das zeitaufwendige Rechnen per Hand und das Arbeiten mit Tabellen überflüssig.

STATIST [1] *(2 Disketten)*
SD 32a/b *DM 30.-*

Sonderdisk-Bestellung

Sonderdisks können Sie telefonisch oder schriftlich bestellen, oder nutzen Sie einfach die Bestellkarte im Heft.

Bei Nachnahme zzgl. DM 4.- Gebühr, Versandkosten DM 5.- (Ausland DM 10.-)

MAXON Computer
Schwalbacher Str. 52
W-6236 Eschborn
Tel: 06196/481811

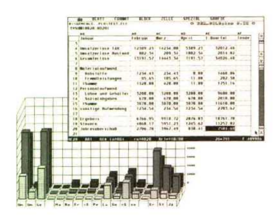


Special Paint 2

Grafik de Luxe

Grafikprogramm der Extraklasse. Neben den vielen nützlichen Funktionen zeichnet sich Special Paint vor allem durch seine Geschwindigkeit, seine bequeme Bedienung und seine Kompatibilität zu bekannten Malprogrammen aus. Special Paint bietet umfangreiche Blockfunktionen, Lasso, superschnelle Lupe, Maskierungen, Clippen, schnelle Bieg-, Zerr- und Drehoptionen, Animation und vieles mehr. Clipboardunterstützung, umfangreiche Textfunktionen (ladbare Fonts, Blocksatz, Zeilenumbruch).

Special Paint [1]
SD 21 *DM 20.-*



GEM-CALCplus 3.0

Tabellenkalkulation

Überall dort, wo mit Zahlen hantiert wird, sei es zur betriebswirtschaftlichen Kostenrechnung, statistischen Auswertung von Meßreihen oder zur Erfassung der eigenen Finanzen, findet ein Kalkulationsprogramm seinen Einsatz. GEM-CALCplus ist ein flexibler und sehr leistungsfähiger Vertreter dieser Kategorie. Neben zahlreichen mathematischen und statistischen Funktionen bietet es eine exzellente Grafikausgabe der Daten als Kuchen-, Linien-, Balken-, Stapel-, Säulen-, Block- und Flächengrafik.

Funktionen und Operatoren:
+, -, *, /, PI, DAT, ABS(), INT(), RND(), LOG(), EXP(), CLG(), SQR(), SIN(), COS(), TAN(), ASN(), ACS(), ATN(), FAK(), NUN(N:n), SUM(), AVE(), STA(), STD(), MUL(), MIN(), MAX(), QMW(), QMN()

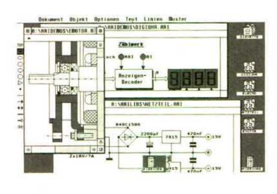
GEM_CALCplus ist eine Weiterentwicklung des weitverbreiteten GEM_CALC (PD)

Die Erweiterungen:
• Arcussinus und Arcuscosinus • Blatt und Block schützbar • Fehlermeldungen mit Erläuterung • Suchfunktion • verbesserte Grafikdarstellung, Grafikausdruck und Grafik-Datenauswahl (Block) • flexible Speicherverwaltung • fixierbare Spalte • erhöhter Eingabekomfort • schnelleres Scrolling • u.v.a.m.

Alte Datenblätter können übernommen werden. (1MB sinnvoll)

GEM-CALCplus 3.0 [1]
SD 44 *DM 25.-*

[1] nur für monochromen Monitor (SM 124)
[2] nur für Farbmonitor



ARIADNE

ARIADNE ist ein objektorientiertes Zeichenprogramm, d.h. Objekte können auch im Nachhinein ohne Auflösungsverlust verändert werden. Es bietet die Möglichkeit, jedes beliebige Grafikobjekt (mit Doppelklick) zu öffnen, worauf eine neue Zeichenebene bereitgestellt wird. Die Objekte auf dieser Ebene können dann wiederum geöffnet werden usw. Diese hierarchische Struktur eignet sich besonders zur Darstellung komplizierterer Dinge, z.B. Blockschaltbilder, Schaltungen etc.
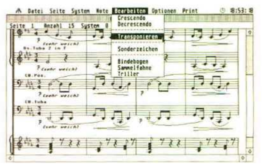
ARIADNE [1]
SD 8 *DM 15,-*

COMPLEX

Quiz

Quiz mit über 3500 Fragen aus den Wissensgebieten Geschichte, Geographie, Sport, Allgemeinbildung, Tierwelt, Kunst, Naturwissenschaft und Theater. Das Programm kann mit eigenen Fragen erweitert werden, somit steht die Möglichkeit zur Schaffung eines spezialisierten Quiz' (z.B. Motorwelt, Jura, Computerkunde oder gar Fremdsprache) offen. (1MB, 1-6 Spieler)

COMPLEX [1]
SD 47 *DM 20.-*



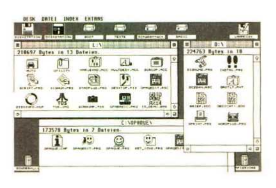
TRISTAN

Notensatzsystem

Für alle Musikfreunde, die nicht nur vom Blatt spielen, sondern auch aufs Blatt schreiben, bietet das Notensatzsystem TRISTAN die ideale Möglichkeit, ihre Noten professionell zu Papier zu bringen. Es lassen sich Partituren mit bis zu 100 Seiten mit max. 32 Notensystemen je Seite bearbeiten. Alle im klassischen Notensatz gebräuchlichen Zeichen lassen sich bequem mit der Maus edieren. Ebenfalls stehen mehrere Notenschlüssel, Sammelfahnen, Triller und Bindebögen zur Verfügung. Automatische Transponierfunktion. Ausdruck auf 9- und 24-Nadeldruckern, im 24-Nadelmodus in *maximaler* Druckerauflösung.

TRISTAN
SD 24 *DM 25.-*

Sonderdisks unterliegen trotz des niedrigen Preises einem Copyright.



OPAQUE

Das Desktop mit neuem Gesicht

Wie wäre es mit einem zweckmäßigen und originellen Desktop? Opaque bietet die Möglichkeit, jedem Programm ein eigenes, sinnbezogenes Icon zuzuordnen. Auch die Laufwerke lassen sich ändern. Weiterhin kann man die Icons mit Wildcards definieren. Samt Icon-Editor und über 100 Icons.

OPAQUE
SD 22 *DM 15.-*

DER MOTOR

Der Motor erklärt mit zahlreichen Grafiken die Funktionsweise eines Verbrennungsmotors. Sehr anschaulich sind die bewegten Grafiken. Das gezeigte Wissen wird zusätzlich in einem Quiz abgefragt. Mit geregeltem Katalysator!!

DER MOTOR [1]
SD 20 *DM 15.-*

Programmierer aufgepaßt!!

Haben Sie nicht auch ein Programm geschrieben, das in diese Serie paßt? Sonderdisketten enthalten leistungsstarke Programme aus allen Bereichen zu günstigen Preisen. Als Autor erhalten Sie eine attraktive Umsatzbeteiligung. Lassen Sie doch mal was von sich hören.

MAXON Computer
Idee Sonderdisk
Industriestr. 26
W-6236 Eschborn

Weitere Sonderdisks

Nr.	Titel	Preis
01	TOS 1.0	nicht mehr lieferbar
02	RCS 1.4	15,-
03	Extended VT52 [1]	15,-
04	Lovely Helper	15,-
05	Accessories	15,-
06	NIKI [1]	15,-
07	VirusEx	15,-
09	Legende [2]	15,-
10	Quinemac [1]	15,-
12	MagicBox ST	15,-
13	Robotwar [1]	15,-
16	Easy Adress [1]	15,-
17	IconDesign	15,-
19	MAKI [1]	15,-
26	Hauskasse [1]	15,-
28	Master Etikett [1]	15,-
30	Würfelpoker	15,-
31	EasySTat [1]	25,-
33	Ultra-Disk	15,-
38	ST-Himmel [1]	20,-
34	Fußball [1]	15,-



SONDERDISK

Sonderdisks beinhalten Programme aus den verschiedensten Bereichen (z.B. Utilities, Grafik, Schulung, Spiele). Sonderdisks ermöglichen den Usern, qualitativ hochwertige Software zu einem kostengünstigen Preis zu erhalten. Im Preis ist eine Beteiligung der Autoren enthalten.

In der nächsten ST-Computer lesen Sie unter anderem

CeBIT '91

Es ist wieder so weit. Vor den Toren Hannovers findet auch in diesem Jahr wieder die CeBIT, die größte Computer-Messe der Welt, statt. Tausende von Ausstellern aus aller Herren Länder präsentieren ihre neuesten Produkte. Wer nicht „live“ dabei sein kann, darf sich auf unseren umfassenden Messebericht in der nächsten Ausgabe freuen.

BASIC auf dem TT

Lange Zeit gehörte Omikron.BASIC zum Lieferumfang des Atari ST, mit den neuen Rechnern Mega STE und TT wird es nicht mehr ausgeliefert. Die Firma Omikron saß jedoch nicht still und hat ein Omikron.BASIC für den TT entwickelt. Doch auch GFA Systemtechnik läßt nicht auf sich warten und arbeitet auf Hochtouren an der TT-Version. Wie die neuen BASICs aussehen und welche Änderungen die Sprachen erfahren haben, lesen Sie in der Mai-Ausgabe der ST-Computer.

Disketten-Tools

Von der Firma Richter wird auf der CeBIT das Programm „Imagic Wizard“ vorgestellt. Gerüchte besagen, daß dieses Programm das PC-Tools für den ST sei. Grund genug für uns, das Produkt ausführlich zu testen. Doch nicht nur Imagic Wizard wollen wir Ihnen vorstellen. Auch Berichte über „Diskus“ von CCD in der neuen Version 2.0 und „Xenon“ von Atari Schweiz finden Sie in Ausgabe 5.

Diskettenlaufwerke im Zehnerpack

Als die Atari STs erschienen, wurden sie zunächst nur mit einseitigen 3,5"-Laufwerken ausgeliefert. Im Gegensatz zu PC-Floppy-Controllern, die in aller Regel bis zu vier Laufwerke gleichzeitig betreiben können, mußte bei den STs auf unkomfortable „Floppy-Switches“ zurückgegriffen werden. Wir stellen Ihnen eine Zusatz-Hardware vor, die nahezu beliebig viele Anschlußmöglichkeiten für Diskettenlaufwerke am ST eröffnet.

Die nächste ST-Computer erscheint am Fr., dem 26.04.91

Fragen an die Redaktion

Ein Magazin wie die ST-Computer zu erstellen, kostet sehr viel Zeit und Mühe. Da wir weiterhin vorhaben, die Qualität zu steigern, haben wir Redakteure eine große Bitte an Sie, liebe Leserinnen und Leser: Bitte haben Sie Verständnis dafür, daß Fragen an die Redaktion nur **donnerstags von 14^{00}-17^{00} Uhr** unter der Rufnummer 06196/481814 telefonisch beantwortet werden können.

Natürlich können wir Ihnen **keine** speziellen Einkaufstips geben. Wenden Sie sich in diesem Fall bitte an einen Fachhändler. Wir können nur Fragen zur ST-Computer beantworten.

Vielen Dank für Ihr Verständnis!

Impressum ST Computer

Chefredakteur: Harald Egel (HE)

Redaktion:
Harald Egel (HE)
Joachim Merz (JM)
Dieter Kühner (DK)
Martin Pittelkow (MP)

Redaktionelle Mitarbeiter:
C.Borgmeier (CBO)
Claus Brod (CB)
Ingo Brümmer (IB)
Derek dela Fuente (ddF)
Stefan Höhn (SH)
Raymund Hofmann (RH)
Claus P. Lippert (CPL)
Thorsten Luhm (thl)
Chr. Schormann (CS)
U.Seimet (US)
R.Tolksdorf (RT)
Thomas Werner (TW)

Autoren dieser Ausgabe:
H.Albrecht
D.Brockhaus
M.Chakravarty
J.Funcke
U.Hax
A.Hollmann
P.Holzwarth
S. Krüppel
Dr.V.Kurz
M.Schneider
O.Scholz
T.Schweitzer
S.Slabihoud
R.Stelljes

Auslandskorrespondenz:
C.P.Lippert (Leitung), D.Dela Fuente (UK)

Redaktion: MAXON Computer GmbH
Postfach 59 69
Industriestr. 26
6236 Eschborn
Tel.: 0 61 96/48 18 14, FAX : 0 61 96/4 11 37

Verlag: Heim Fachverlag
Heidelberger Landstr. 194
6100 Darmstadt 13
Tel.: 0 61 51/5 60 57, FAX : 0 61 51/59 10 47 + 5 60 59

Verlagsleitung:
H.J.Heim

Anzeigenverkaufsleitung:
U.Heim

Anzeigenverkauf:
K.Sterna, H. Arbogast

Anzeigenpreise:
nach Preisliste Nr.6, gültig ab 2.1.91
ISSN 0932-0385

Layout:
Manfred Zimmermann (vtl.)

Titelgestaltung:
Axel Weigend

Fotografie:
Andreas Krämer

Illustration:
Manfred Zimmermann

Produktion:
B.Kissner

Druck:
Frotscher Druck GmbH

Lektorat:
V.Pfeiffer

Bezugsmöglichkeiten:
ATARI-Fachhandel, Zeitschriftenhandel, Kauf- und Warenhäuser oder direkt beim Verlag

ST Computer erscheint 11 x im Jahr
Einzelpreis: DM 8,-, ÖS 64,-, SFr 8,-
Jahresabonnement: DM 80,-
Europ. Ausland : DM 100,- Luftpost : DM 130,-
In den Preisen sind die gesetzliche MWSt. und die Zustellgebühren enthalten.

Manuskripteinsendungen:
Programmlistings, Bauanleitungen und Manuskripte werden von der Redaktion gerne angenommen. Sie müssen frei von Rechten Dritter sein. Mit seiner Einsendung gibt der Verfasser die Zustimmung zum Abdruck und der Vervielfältigung auf Datenträgern der MAXON Computer GmbH. Honorare nach Vereinbarung. Für unverlangt eingesandte Manuskripte wird keine Haftung übernommen.

EIN GUTER FREUND

»Mortimer ist ein wirklich gelungenes Programm, das man jedem ans Herz legen kann.«
PD-Journal 8/90, S. 26

»Die Firma OMIKRON hat sich offenbar ganz am Endbenutzer orientiert, und das hat zu einem wirklich guten Ergebnis geführt.«
XEST (österreichisches ATARI-Magazin) 2/90, S. 18

»... ein Butler, von dem man sich wirklich gern verwöhnen läßt.«
ST-Magazin 5/90, S. 21



MORTIMER PLUS

Für viele unserer Kunden ist Mortimer ein guter Freund geworden. Er war stets da, wenn er gebraucht wurde; verstand sich gut mit allen anderen Programmen – und packte immer kräftig mit an. In diesem Jahr hat er nochmals kräftig dazugelernt. Und ist so – wie wir meinen – ein noch besserer Freund geworden. Näheres erfahren Sie im Prospekt oder telefonisch.

Mortimer Plus DM 129,– Upgrade DM 60,–

Mortimer DM 79,–

NEUHEITEN

+ Texteditor mit automatischem Zeilenumbruch, Blocksatz und Menüzeile
+ Speichermonitor: Daten retten nach Absturz beliebiger Programme
+ Dateiauswahlbox ins Betriebssystem eingebunden
+ erweiterter Tastaturmakro-Treiber
+ lauffähig auf ATARI TT
+ Uhrzeit einstellen & über Kaltstart retten
+ trotzdem weniger als 80 Kbyte – kein Problem selbst für einen 520 ST

Mortimer Plus kann natürlich alles, was Mortimer kann – und das ist eine ganze Menge.

OMIKRON.Soft + Hardware
Sponheimstr. 12a · D-7530 Pforzheim
Telefon 0 72 31 / 35 60 33



XEST, Webgasse 21, A-1060 Wien
OMIKRON. France, 11, rue dérodé, F-51100 Reims
Elecomp, 11, avenue de la gare, L-4131 Esch/Alzette
Jotka Computing, Postbus 8183, NL-6710 AD Ede